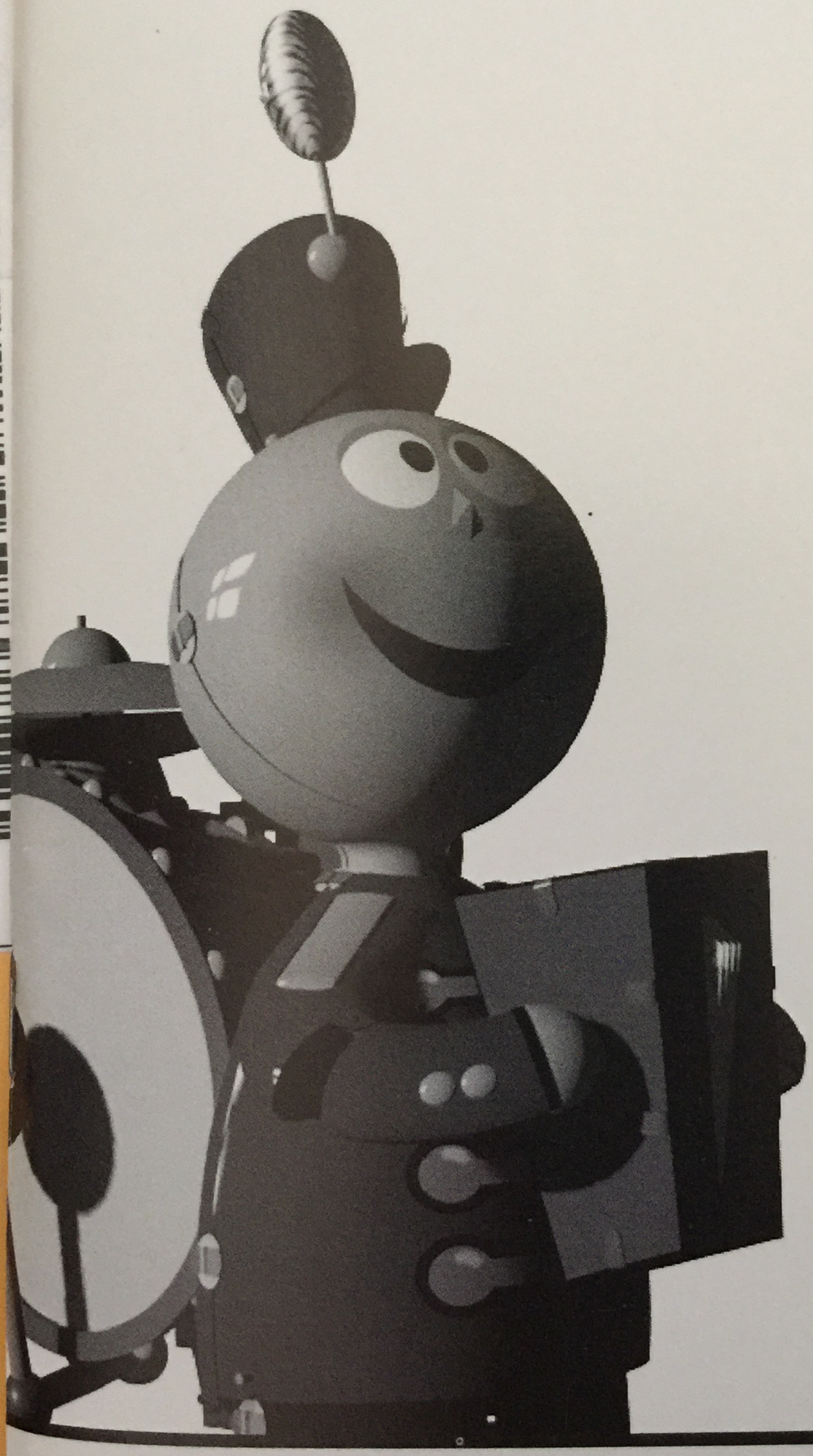
MacRenderMan



P·I·X·A·R

MacRenderMan®
Version 1.3.2

User's Guide

August, 1993

Pixar
1001 West Cutting Blvd.
Richmond, CA 94804

Copyright © 1993 Pixar
All Rights Reserved
Patented: US 4,897,806; UK 2,177,577; CANADA 1,256,224. Patents Pending.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of Pixar.

The information in this book is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Pixar. Pixar assumes no responsibility or liability for any errors or inaccuracies that may appear in this book

The software described in this book is furnished under license and may only be used or copied in accordance with the terms of such license.

Pixar®, RenderMan®, and the Pixar logo are registered trademarks of Pixar. PhotoRealistic RenderMan™, RIB™, and Reyes™ are trademarks of Pixar. Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Pixar was aware of a trademark claim, the designations have been printed in initial caps or all caps—for example, Macintosh, CLARIS.

For units of the DoD: Restricted Rights Legend

Use, duplication or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at 252.227-7013.

For civilian agencies: Restricted Rights Legend

Use, reproduction or disclosure is subject to restrictions set forth in subparagraph (a) through (d) of the Commercial Computer Software-Restricted Rights clause at 52.227-19 and the limitations set forth in Pixar's standard commercial agreement for this software. Unpublished - rights reserved under the copyright laws of the United States.

Pixar, 1001 West Cutting Blvd., Richmond, California, 94804 USA

TABLE OF CONTENTS

1 Introduction to MacRenderMan 1

- **Basic Concepts — Modeling, Rendering and Shading** 2
  - Modeling vs. Rendering 2
  - Modelers and Scene Description 3
  - Renderers and Shading 4
  - Summary 5
- **The MacRenderMan Software** 6
  - RenderMonitor 6
  - Chooser 6
  - PhotoRealistic RenderMan Renderer 7
  - Vector RenderMan Renderer 7
  - RenderApp 7
- **The Pipeline Through the MacRenderMan Software** 8
- **Rendering and the "Print-Queue" paradigm** 8
- **Customizing MacRenderMan** 9

2 RenderApp 11

- **The RenderApp Window** 11
- **The File Menu** 12
  - The Save Dialog 14
  - Notes on saving 17
- **The Edit Menu** 17
- **The Image Menu** 18
- **The Render Menu** 21
  - Set Crop Window 22
  - Render Setup... using PhotoRealistic RenderMan 23
  - Render Setup... using Vector RenderMan 26

Render... (to a window) 28
Render To File... 30
Rendering large images 33

3 **Rendering RIB Files** 35

Managing Rendering Jobs with RenderMonitor 35
Background Rendering 36
Bringing up the RenderMonitor Window 36
Waiting job status messages 39
Messages displayed when a job wouldn't render 40
Setting the rendering time 41
The RenderMonitor File Menu 42
The Preferences dialog 43
RenderMonitor Log Files 44
The RenderMonitor Edit Menu 44
The RenderMonitor Jobs Menu 45
Problems reported by RenderMonitor 46
Warnings 46
Errors 47
Using the Chooser 47
Arranging the order of renderers 49
Renderer configuration options 49

APPENDIX: Editing RIB Files 53

RIB Syntax 54
Header Information 55
Light Information 55
Surface Information 56
Texture Creation 56

Index 59

1 Introduction to MacRenderMan

Before starting to play with the world of possibilities this package of MacRenderMan software represents, take the time to read this introduction. Although MacRenderMan is designed to be very easy — if not altogether transparent — to use, it has powerful capabilities that might quickly become overwhelming without some background.

The idea behind the MacRenderMan software is to make the act of generating a high-quality computer image as easy as printing a document. Most of its behavior is automatic, but you are given every opportunity to get your hands dirty when you feel the urge.

In this chapter we will go over some of the basic concepts of computer image synthesis as an introduction to the terminology or as a refresher to those familiar with it. This is followed by an overview of the software.

Basic Concepts — Modeling, Rendering and Shading

Chances are you have a picture in your mind that you would like to see as a computer image. As with any task involving computers, your ability to do this is dependent on your ability to describe the task with precision and accuracy.

Modeling vs. Rendering

Modeling is the process of setting up a scene description. It describes the precise geometry, placement and appearance of all the objects in the scene, the characteristics of all light sources illuminating the scene, and the characteristics of the viewer of the scene.

Rendering is the process of generating a visual image from a precise description of a scene.

These two processes are performed by programs known as *modelers* and *renderers* respectively.

The RenderMan Interface is a language for *modelers* to communicate with *renderers*. MacRenderMan handles the details of moving information between modelers and renderers on the Macintosh. It provides standardized rendering services to the variety of modelers on the market that support the RenderMan interface, as well as a front end to its own renderers and other renderers on the Macintosh.

In order to use MacRenderMan, the files generated by a modeler and processed by a renderer must be in a format known as the *RenderMan Interface Byte-stream* protocol, or *RIB*, and are known as *RIB files*. Although you may never need to know the details of this file format, it is useful to understand what it is and the type of information it deals with. We touch on this next.

Modelers and Scene Description

A scene described by a modeler requires a precise specification of the following elements:

- objects (geometry and appearance)
- light sources
- viewer

The *objects* in a scene are described by their *geometry* (the shapes of the objects and the relative placement of the objects in space) and by the *appearance* of their surfaces. Describing an object's shape is often the most involved task performed by a modeler. It must be built up from simpler shapes into a coherent form. The *appearance* of surfaces is determined not only by the color of the objects but also by more complex optical properties such as the *opacity*, *reflectance*, and *texture*. Different modelers will have varying facilities for describing these attributes.

The *lighting* in a scene is described by specifying the location and characteristics of every light source used to illuminate the objects. A modeler may have some facilities for placing and describing several kinds of light sources in a scene, such as the *ambient* light in the scene, *distant light sources*, *spotlights*, etc.

Describing the *viewer* (or *camera*) involves not only the location of the viewer relative to the objects in the scene and the direction the viewer is facing, but also a number of viewing characteristics the modeler may control with names such as *field of view*, *depth of field*, *f-stop*, and *perspective*.

All of these concepts should be described in more detail in the documentation provided by the modeler you are using to create the scene. We mention them here only because these

are the kinds of information passed in RIB files from a modeler to a renderer.

Renderers and Shading

Once a scene description has been produced with the aid of a modeler, a renderer performs the job of generating an image of the scene.

The details of the rendering process are not of much concern to us at this point. Steps with names such as 3-D transformations, hidden surface removal, sampling, anti-aliasing, exposure, imaging, quantization, and other computations, are performed automatically by the renderer and you may never have occasion to know about them at all. Nevertheless, there are ways to control these processes if you want to. One example that offers a lot of control over the rendering process is *shading*.

Shading is that part of the rendering process that calculates the exact appearance of every surface in the image.

A *shader* is a program that specifies the process used to calculate the appearance of an element in a scene. Shaders are called by a modeler but executed by the renderer. A shader can be applied to an object in the scene to determine the color, reflectivity, and texture of its surfaces; to a light source to describe the color, direction, and intensity of light hitting surfaces from the light source; or to introduce some atmospheric effect such as fog, which affects the behavior of all the light in a scene.

In the RenderMan environment, shaders exist as programs written in the RenderMan Shading Language. These are used by the renderer whenever they are invoked by name in the scene description.

Summary

In general, you use a modeler to tell a renderer what to do (i.e., what the image should look like). The modeler has facilities for describing the main elements of the scene so they will appear the way you wish. The renderer handles the details of image generation for you.

For this reason, you will spend most of your time working closely with a modeler to describe your scenes, while the renderer works in the background, generating the images you've requested. Rendering becomes an almost invisible process, with images appearing automatically with no further action on your part.

Nevertheless, there may come a time when you want to get involved in the details of rendering and do things not addressed by the modeler you are using. You will then want to explore that space between modeling and rendering, and this is what RenderMan is all about.

Take shaders, for example. You invoke them, even though you may not be aware of them, whenever you use your modeler to describe the surface appearance of an object, or to specify a light source or any of a number of special effects your modeler may offer that affect how things are to appear. The appropriate shaders are located automatically by the renderer and the requested effect appears in the final image.

So, you can actively control shading and other details of the rendering process at different levels, depending on how ambitious and "advanced" you want to get.

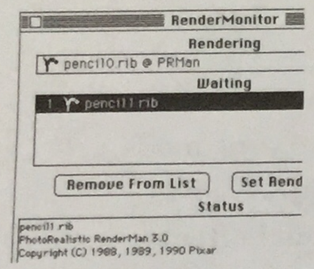
The MacRenderMan Software

MacRenderMan is a package of applications designed to bridge the gap between modelers and renderers on the Macintosh. The package comes with its own renderer: Pixar's PhotoRealistic RenderMan renderer.

Scene descriptions are created by a modeler in the form of a RIB file, which is then passed to a renderer, which in turn generates an image that can be saved or displayed on your Macintosh.

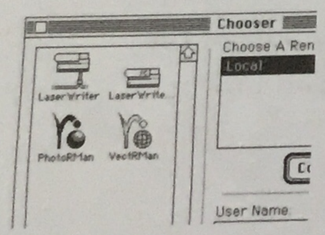
The applications provided in the MacRenderMan package are intended to make using this pipeline as simple as possible.

RenderMonitor



Similar to the PrintMonitor desktop application available under MultiFinder, RenderMonitor allows you to render images in the background, so you can continue with other functions while it works. It also manages the queueing of renderer requests and allows you to examine this queue and make changes to it.

Chooser

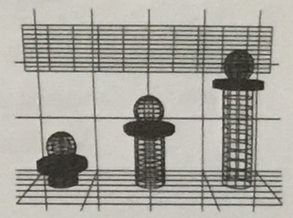


The standard Macintosh Chooser desk accessory allows you to select a renderer from among those that are available, and to set up renderer options for each one, if necessary.

PhotoRealistic RenderMan Renderer

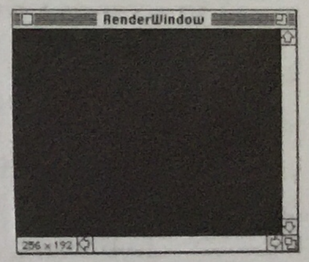
The MacRenderMan package includes Pixar's high-quality renderer, known as the PhotoRealistic RenderMan renderer (or *PhotoRMan*), as the renderer of choice for most applications. This is the default renderer selected by the Chooser and is based on Pixar's Reyes algorithm.

Vector RenderMan Renderer



A second renderer, known as the Vector RenderMan renderer (or *VectRMan*), is also included with the MacRenderMan package to generate wireframe images as quick approximations of more detailed renderings.

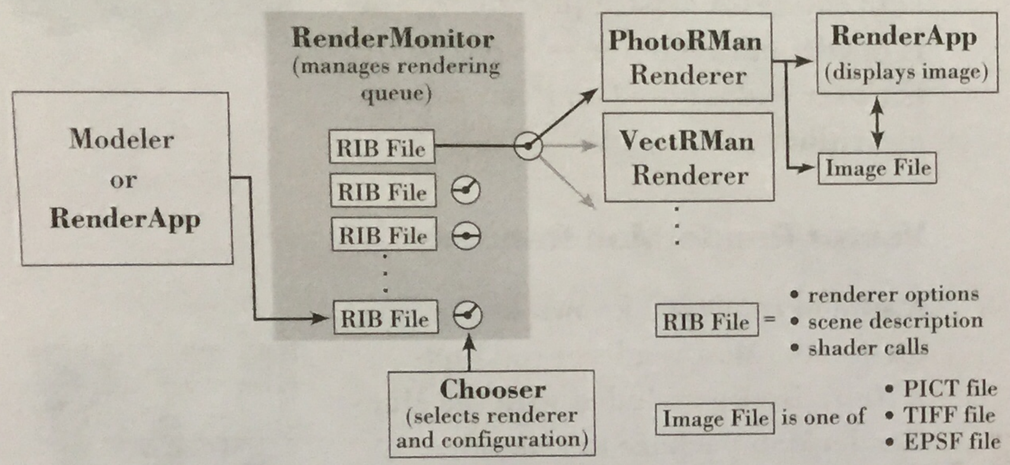
RenderApp



RenderApp is a desktop application that allows rendered images to be displayed on a color Macintosh monitor. It can also be used to display and save image files in PICT format. The renderer can then use these files to generate *texture maps*. RenderApp also has the capability to queue up jobs for rendering, as well as configure the available renderers.

The Pipeline Through the MacRenderMan Software

Here is a diagram that shows how rendering might typically proceed:



Rendering and the "Print-Queue" paradigm

The idea behind the design of the MacRenderMan package is to make rendering as straightforward and transparent a process as printing a document. It might be useful to think of the analogies between rendering in the RenderMan world and printing in the PostScript world:

- The RenderMan interface is analogous to the PostScript language, which acts as a standard interface language between a number of document preparation programs and PostScript-processing printers.
- The RenderMonitor is analogous to the PrintMonitor application familiar to users of MultiFinder. Requests are spooled and processed automatically in the background.

- A modeler is analogous to a document preparation program or editor that generates page descriptions in the form of PostScript files.
- A renderer is analogous to a printer that takes a stream of PostScript code and generates a page of text or graphics. Just as you could have more than one printer available, you could also have more than one renderer available, depending on your needs and/or time constraints. Hence the need for the Chooser to allow you to select the most appropriate renderer for the job.
- A shader is analogous to a font in PostScript. Just as fonts can be applied to different words or other elements on a page, shaders can be applied to different objects or other elements in a scene. Libraries of shaders can be built up just as there are libraries of fonts that can be used when composing a page to be printed.

Customizing MacRenderMan

We have provided a text file that allows you to customize various aspects of the rendering process. These are things like where temporary display files should go, where applications should look for texture files and Looks, etc. This file is *Pixar Preferences*, and is in the System folder's *Preferences* folder. You can open it with any word processor and edit its contents as you wish. The file is self-documenting — it contains comments that tell you how to edit the various options.

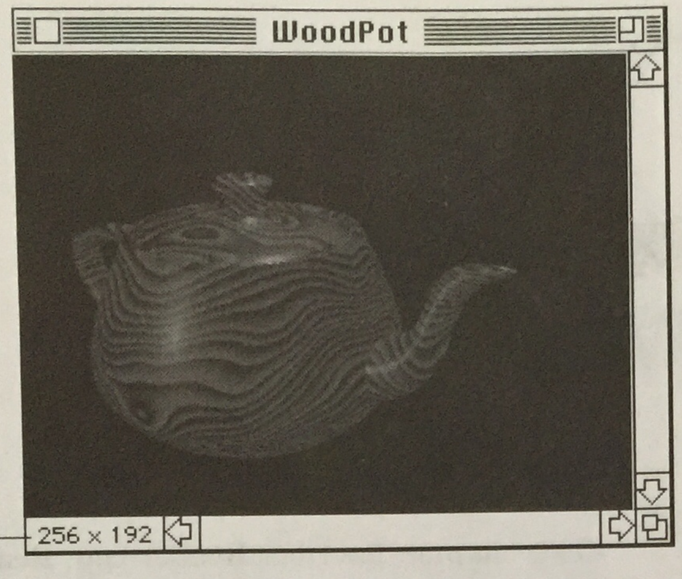
2 RenderApp

This chapter describes RenderApp. RenderApp is an application that sends RIB files to a renderer and displays rendered images in a window. It also serves as a way to manipulate images in some simple ways. You can use RenderApp independently of any other application.

For information on choosing a renderer and monitoring the rendering process see the next chapter.

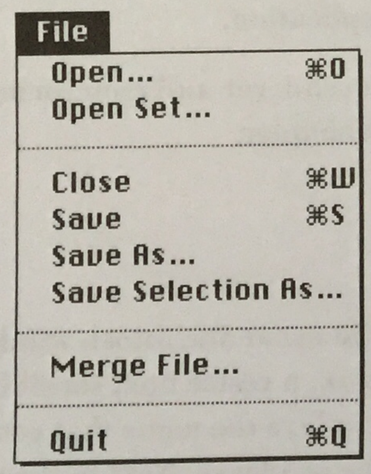
The RenderApp Window

The RenderApp window has the usual Macintosh window features: a close box, a zoom box, a resize box, scroll bars, and a title bar. The title bar displays the name that gets put into the RIB file that goes to the renderer. Next to the name is the zoom factor at which the image is currently being displayed. As in any other Mac window, the zoom box either expands the window to its maximum allowable size or shrinks it back to its former size. When you open or render an image, the image automatically gets displayed at a magnification that allows the window to fit on the screen. The image size, in pixels, appears in the lower left corner of the window.



Maximum window size

The File Menu



Open.... This allows you to select an image file to display in a RenderApp window. Only PICT and TIFF files are recognized as image files.

Open Set.... This allows you to open a set of image files to display as a single image in a RenderApp window. A set consists of a series of image files, each with a sequential three-digit suffix, for example *Image.pict 000*, *Image.pict 001*,

Image.pict 002, etc. The first filename in the set must end with *000* or *001* and these will be the only files available to select for opening. When you select the first file in the set, RenderApp opens the entire set. In this way a single image can be constructed from a set of rendered crop windows.(-See "Rendering large images" on page 33 for information on rendering an image as a set of crop windows.)

Close. This closes the current window after checking to see if you want to save the image to a file.

Save. If the current image has not previously been saved, this item works just like Save As... (see below). Once the current image has been saved, this option resaves the image to the same file. If you are saving a number of windows each of which is a subsection of one large image to be assembled using Open Set, be sure to use names with sequential three-digit suffixes (for example *Image.pict 000*, *Image.pict 001*, *Image.pict 002*, etc.). See below for an explanation of the dialog box.

Save As.... This brings up a dialog that allows you to save the file with a different name, in a different folder, or on a different disk. This also allows you to specify the details of how the image is to be saved. If you are saving a number of windows each of which is a subsection of one large image to be assembled using Open Set, be sure to use names with sequential three-digit suffixes (for example *Image.pict 000*, *Image.pict 001*, *Image.pict 002*, etc.). See below for an explanation of the dialog box.

Save Selection As.... This is the same as Save As except that you can save a rectangular subsection of the image by clicking and dragging the mouse to enclose the desired area, and then using this menu item. See below for an explanation of the dialog box.

Merge File.... This brings up a dialog box that allows you to choose a PICT or TIFF image to merge on top of the currently displayed image. The transparent parts of the new image file (e.g., the black background pixels in the WoodPot image), allow parts of the original image to show through. You can use this, for example, to construct several images, all with the same background but with changes to the objects in the foreground. This can also be used to reassemble sections of an image created using crop windows into one image. If the imported image is bigger than the one in the window, the window will expand to accommodate the new image.

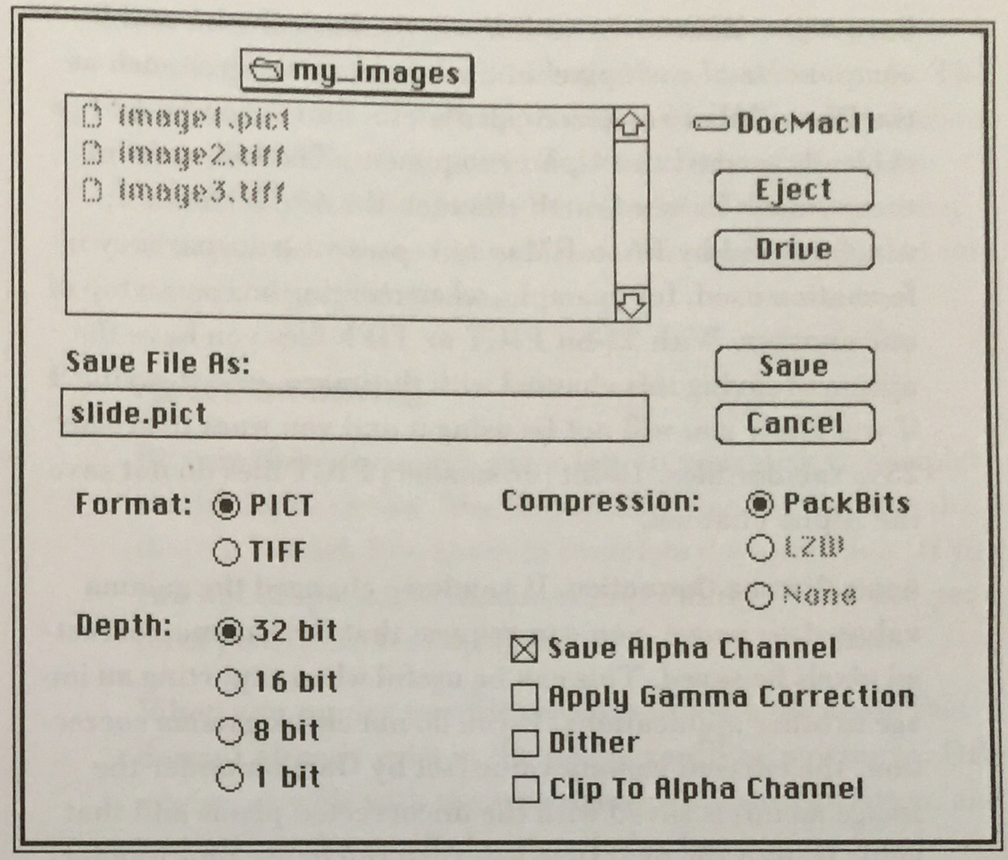
Quit. If all current windows have already been saved, this closes RenderApp. If a window has not been saved, a dialog box appears asking if you'd like to save it. If so, the Save dialog box comes up, allowing you to save in the usual way. If not, RenderApp closes.

The Save Dialog

The Save, Save As and Save Selection As menu items bring up the same dialog box. The top half of the dialog box contains controls that allow you to select a folder and to type in a file name for the image file. The bottom half of the dialog specifies how the image is to be saved.

Format. The image can be saved as either a PICT or a TIFF (Tagged-Image File Format) file.

Note: The TIFF format is a broadly-defined format. This means that there are different sorts of valid TIFF files. An application that imports TIFF may not be able to read all sorts of TIFF. MacRenderMan TIFF files can be read by many applications, but not by all. If you have a problem reading a MacRenderMan TIFF file into another application, try resaving the image with a different set of options:



32-bit with no Alpha channel, for example. If you have an application that will not read any kind of MacRenderMan TIFF, please notify Pixar.

Depth. This indicates the number of bits used to represent a pixel. PICT files can be saved at any depth. TIFF files can only be saved at 32 bits/pixel.

Compression. PICT files are compressed using PackBits compression. TIFF files can be saved using PackBits or LZW compression, or without any compression at all. Which compression scheme you choose may depend on the format(s) supported by an application to which you export the image.

Save Alpha Channel. In addition to the Red, Green, and Blue components of each pixel in the image, a renderer such as the PhotoRMan renderer calculates a fourth component generically termed the *Alpha* component. The collection of these values form a fourth channel, the *Alpha channel*, which is used by PhotoRMan to represent transparency information used, for example, when merging images on top of one another. With 32-bit PICT or TIFF files you have the option of saving this channel with the image, or not saving it if you know you will not be using it and you want to create 25% smaller files. 16-bit (or smaller) PICT files do not save the Alpha channel.

Apply Gamma Correction. If you have changed the gamma value of an image, you can request that the gamma-corrected pixels be saved. This can be useful when exporting an image to other applications. If you do not check gamma correction, the current gamma value (set by Gamma under the Image menu) is saved with the uncorrected pixels and that value is used the next time you read the image into RenderApp.

Dither. When saving a 32-bit image as a 16-bit (or lower) PICT file, by default a process called *dithering* is normally used to reduce color-banding due to the fewer colors available at 16 bits/pixel. If you find that dithering produces undesirable effects for a particular image, you can uncheck this box to turn dithering off. This setting overrides the Dither setting under the Image menu.

Clip To Alpha Channel. Checking this box when creating a PICT file causes the Alpha (transparency) channel to be used as a mask, and a "cutout" image created consisting of only the opaque pixels in the image. The black background pixels with 0 in the Alpha channel are not saved. Not only does this save a considerable amount of space, but also allows the interesting pixels from one image (e.g. a rendered object) to be "cut out" and merged into another image. This is especially useful with applications like Aldus Persuasion, Microsoft PowerPoint, etc. To see exactly what will be saved, display only the Alpha channel of the image using Channels under the Image menu. Whatever is white in this image is what will get saved.

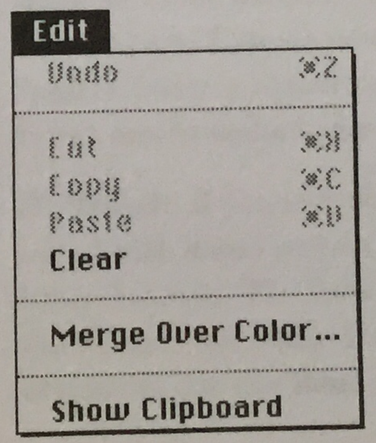
Notes on saving

Be sure there is enough space left on your disk to complete rendering or saving. MacRenderMan does not ensure that there is enough free space to complete the operation. If you run out of space, the Macintosh operating system will perform poorly. To free up space, just delete some files.

When you render to a file, be sure to use a file name that doesn't already exist in the folder you'll be writing to. Otherwise, any file with the same name will be overwritten, and you will not be notified with a dialog box.

The Edit Menu

Only three of these menu items are currently enabled: Clear, Merge Over Color, and Show Clipboard.



Clear. If you have selected an area of an image with the selection box defining a crop window, using Clear fills the selected area with black.

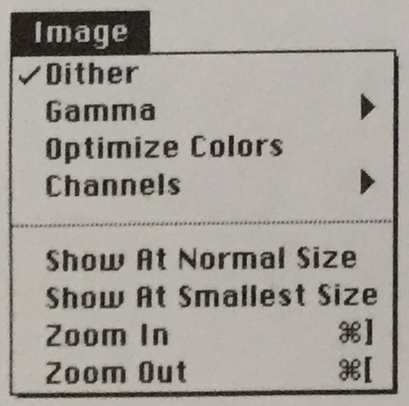
Merge Over Color. This brings up the color picker, allowing you to change the color of the background, as defined by the Alpha channel—once. To see what will change in an image, view just the Alpha channel using the Image menu's Channels item. Whatever is black in this view will be replaced by the color you choose. Once you have changed this color, you cannot change it again: you must close the window and open it up again.

Show Clipboard. This simply brings up the Clipboard.

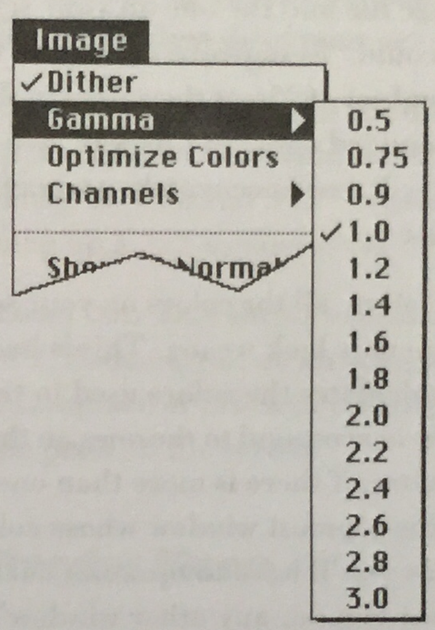
The Image Menu

The Image menu provides control over how an image is displayed on your screen. Gamma is the only setting here that gets saved when you save the image to a file; you will have to reset the others when you redisplay the image. They apply only to the screen, and only for the time the image is displayed.

Dither and Optimize Colors are useful only when displaying in 8-bit mode, and are unavailable otherwise.



Dither. If you have an 8-bit display you may notice discrete bands of colors (or shades of grey) in your image. Turning on Dither will smooth out the transitions between these bands. However, if there are no bands at all, the image may benefit more from having Dither off.



Remember that this affects only how the image is currently displayed on your screen. If you were to save the image in a file you would not be saving an 8-bit dithered image unless you explicitly requested it in the Save As dialog.

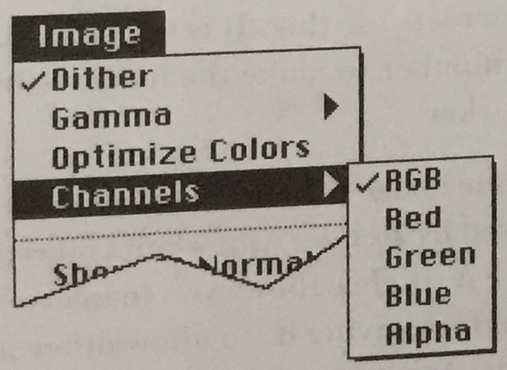
Gamma. Some monitors tend to display more or less "dark" than others. Gamma corrects for this. It is set to 1.0 by default. Choose a higher number to make the image lighter, a lower one to make it darker.

By default, if you save the image the current gamma value is saved with it and restored by RenderApp when you read the image back in. The Save As dialog allows you to instead gamma-correct the image before saving it, to allow other applications to use the RenderApp Gamma.

Optimize Colors. This option improves the accuracy of the colors in your image if you have an 8-bit display. MacRenderMan produces images whose colors are chosen from a palette of millions of colors. But since 8-bit displays allow only 256 colors at a time, the correspondence between the color specified in the image file and the one on your screen may not be as good as it could. To improve the match Optimize Colors uses an optimal set of 256 of the colors in the image, instead of the 256 standard ones. In this way it ensures that the colors displayed will more accurately portray the colors in the original image.

When you use Optimize Colors, all the colors on your screen except those in your image may look wrong. This is because MacRenderMan only recalculates the colors used in the image, which may or may not correspond to the ones on the rest of your screen. Furthermore, if there is more than one RenderApp window up, it's the topmost window whose colors are optimized. This means you'll have to optimize each image individually, and when you do, any other window's colors may look wrong until you bring it to the front.

Channels. This allows you to display all the channels of the image at once, or each one individually. The default is RGB, which displays the Red, Green and Blue channels together in the image.



Show at Normal Size. This causes the image to be displayed at its normal size: each pixel in the image takes up one pixel on the screen.

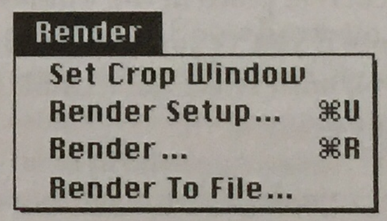
Show at Smallest Size. This zooms out on the image to allow it to be displayed in as small a window as possible. This can be useful when displaying several images at a time.

Zoom In. This magnifies the display of the image by a factor of 2, using pixel replication. For example, zooming in on an image at "Normal Size" causes each pixel in the image to take up a 2x2 square of pixels on the screen.

Zoom Out. This shrinks the display of the image by a factor of 2. Zooming out on an image at "Normal Size" causes each 2x2 square of pixels in the image to be represented by a single pixel on the screen.

The Render Menu

This menu gives RenderApp a capability normally found in a modeler, namely, setting up jobs for rendering. It allows you to create a crop window (a subregion of the display window), to set options for the renderer you've selected in the Chooser, and to render an image to a window or to a file.



A Render Setup dialog identical to the one available here may or may not be available in other modeling applications. It is the interface provided for configuring the renderer from within a modeler.

The Render Setup dialog is different for different renderers. Since MacRenderMan comes with two renderers, PhotoRealistic RenderMan and Vector RenderMan, this section will describe the Render Setup dialog box for each.

Most of the settings available in the dialog boxes associated with this menu are saved across launches and machine reboots. This means that except for the ones listed below, the settings in place at the time you last quit RenderApp will still be in place the next time you start it up. Below are menu items that display dialog boxes containing settings that get reset to a default:

Render Setup... (Esoterica dialog):
Setting: Crop Windows
Default: 0, 0, 1, 1

Render... and Render To File...:
Setting: Merge
Default: (off)
Setting: Name of image to render
Default: Untitled

Set Crop Window

You can use this menu item to rerender just a portion of an image and merge it into the correct place in the window. It will only work for this purpose if you've just rendered to a window, and not a file, and you must select the RIB file from which the image was originally created.

This capability is useful, for example, if you want to use the Vector RenderMan renderer to get an idea of the positions of objects, and then rerender just one object with PhotoRealistic RenderMan to see how it will look. It is also useful if an image requires too much memory or time to render as you can render the whole image at lower quality (or with VectRMan) and then subdivide it up into smaller pieces to render at high quality. To do this, after the image has rendered to a window:

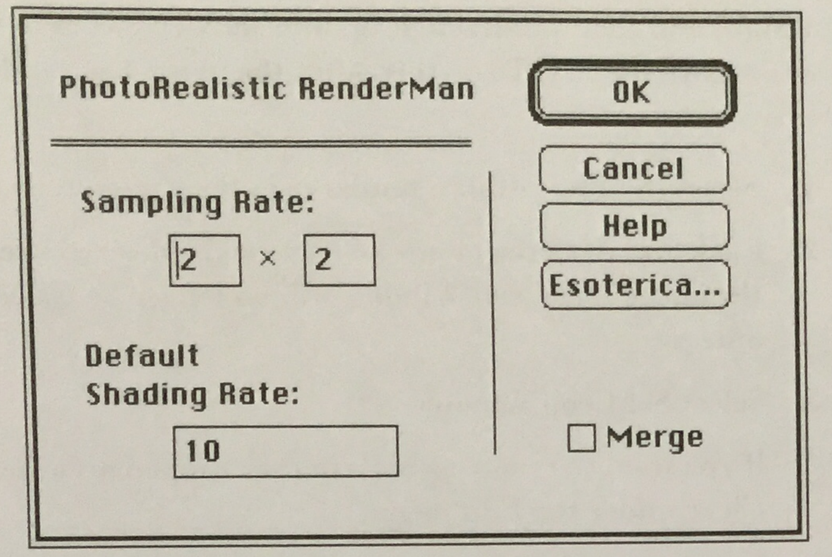
1. Select the PhotoRMan renderer in the Chooser.
2. Click and drag the mouse to enclose the desired area of the image. Set Crop Window will no longer be grayed out.
3. Select Set Crop Window.
4. If you want to remove what's in the cropped area, select Clear under the Edit menu.
5. Select Render and choose the same RIB file used to create the first image.

The Render Setup dialog will, by default, contain the correct Crop Window values to rerender the subregion into the current image window in exactly the right place.

Once you have reset the crop window values, the Set Crop Window menu item becomes Reset Crop Window. When selected, this resets the Crop Window values to their defaults.

Render Setup... using PhotoRealistic RenderMan

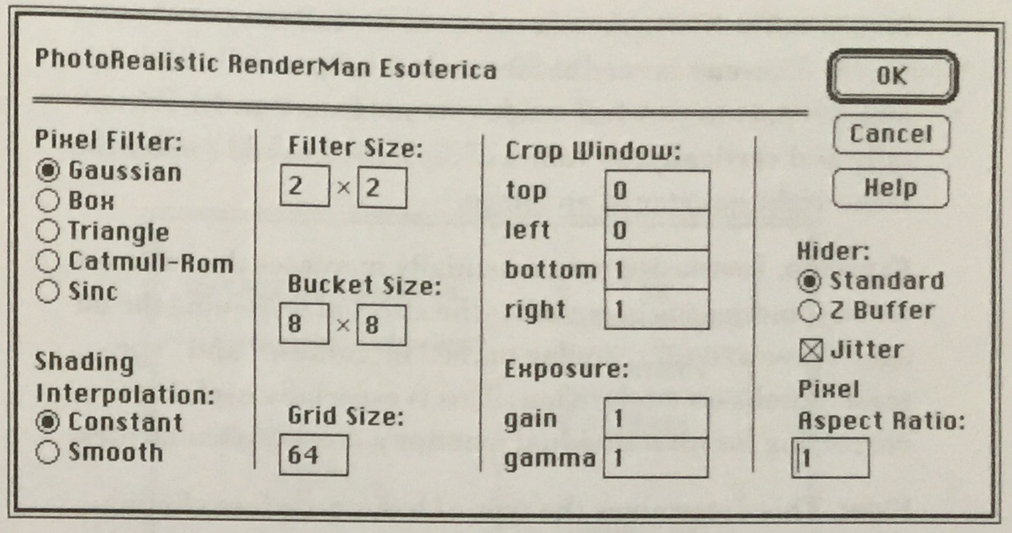
When the PhotoRealistic RenderMan (PhotoRMan) renderer is selected in the Chooser, the Render Setup dialog box sets those options unique to the renderer that control how it is to produce an image. When you change one of its options, the change remains in effect until you reset it. Holding the Option key down while selecting Render Setup sets all the values in the Render Setup dialogs to their default values. These options affect only the file about to be rendered, not files that are already in the queue. These options are overridden by equivalent commands in the RIB file itself.



Sampling Rate. Imagine that each pixel has a grid drawn on it. The renderer looks at the color of the surface below each gridline intersection. It then determines the pixel's color based on what it sees. The Sampling Rate simply sets the number of horizontal and vertical lines in the grid. Higher numbers increase the accuracy of the colors, but increase rendering time accordingly. A useful range is 1x1 to 4x4.

Default Shading Rate. This controls the amount of detail that will appear generally in the image. Your application may allow you to set this for individual objects. If not, all objects will be rendered with this Shading Rate. High values make for faster rendering, but yield less detail, while values close to 1 (or less) increase the level of detail at the expense of rendering speed. Values of 10 or even 100 are useful for rough approximations.

Esoterica.... This button brings up another dialog box, one that allows you to control more exotic aspects of the rendering process. For most purposes you should never have to change these defaults.



Pixel Filter. These are the types of filter available with PhotoRealistic RenderMan. Box is usually appropriate. Gaussian creates a slightly smoother, higher-quality image.

Shading Interpolation. Use Constant in most cases; use Smooth for extremely high quality or when using a high shading rate (expect slower rendering).

Filter Size. Increasing this increases the effectiveness of the Pixel Filter. For higher quality images make this 2x2; 4x4 for *very* high quality (and slower rendering).

Bucket Size. The renderer renders sequential square patches of an image. This sets the size in pixels of an individual patch. Larger buckets increase the rendering speed slightly at the expense of memory use.

Grid Size. This sets the size of a grid related to the bucket size and the shading rate. A good minimum value is the product of the two bucket size numbers divided by the shading rate. So, for example, with a bucket size of 12x12 and a shading rate of 4, the grid size should be at least 36. Large values increase performance at the expense of memory.

Crop Window. This selects a subset of the full image to be rendered. This can be useful for rendering "panels" of a large image separately. A full image extends from 0 to 1 horizontally and vertically; so values of .5, .5, 1, 1 would render the lower right quarter of an image.

Exposure. Increasing gain essentially increases the color intensity; increasing gamma has the effect of lightening the image. These are very similar to the "brightness" and "contrast" knobs on a television. This is especially useful for correcting for an individual monitor's display peculiarities.

Hider. This determines the type of hidden-surface elimination technique to be used. Standard produces higher quality images and longer rendering times, while Z Buffer is better for faster rough approximations.

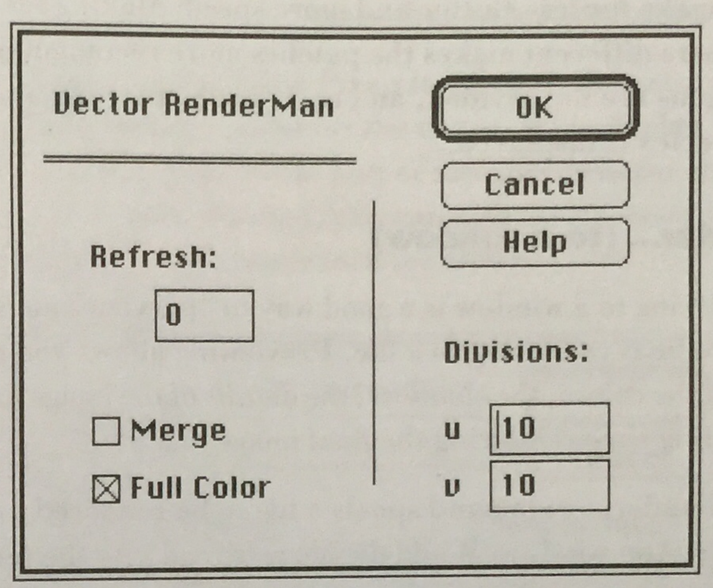
Jitter. When using the Standard Hider and a high sampling rate, turn this on to help prevent aliasing, or "jaggies" along straight lines that lie close to the vertical or horizontal.

Pixel Aspect Ratio. Unless you have a monitor that uses rectangular pixels, leave this set to 1, indicating square pixels.

Render Setup... using Vector RenderMan

When the Vector RenderMan (VectRMan) renderer is selected in the chooser, the Render Setup dialog box sets those options particular to the VectRMan renderer that control how it is to produce an image. When you change one of its options, the change remains in effect until you reset it. Holding the Option key down while selecting Render Setup sets all the values in the Render Setup dialogs to their default values. Since Vector RenderMan renders only lines, the only surface characteristic you will see in the scene is color.

As with any other renderer these options affect only the file about to be rendered, not files that are already in the queue. These options are overridden by equivalent commands in the RIB file itself.



Refresh. This sets the number of vectors that get saved up before being displayed. Low numbers slightly increase rendering time; higher numbers decrease it. Here 0 is a special case: it means to save *all* the vectors before displaying them. This value makes most efficient use of the renderer's time.

Merge. If checked, this simply renders that part of the image defined by the Crop Window settings into the current RenderApp window, merging it on top of any image already there. Any black background in the new image is considered transparent, allowing the pixels from the original image to show through.

Full Color. This toggles between displaying the lines in their respective colors when the box is checked, and displaying them in white when the box is empty.

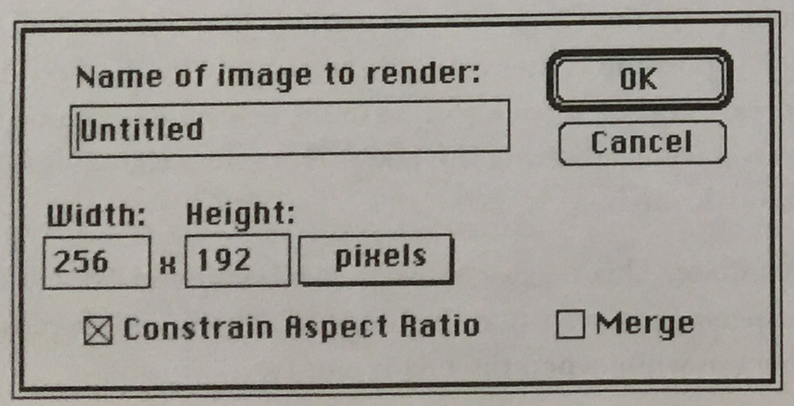
Divisions. The product of these numbers specifies the number of pieces into which a surface gets divided. High numbers force the renderer to spend time filling your screen with more lines, but produce more accurate shapes. Low numbers make for less clutter and more speed. Making the two numbers different makes the patches more rectangular. Polygons are not divided, and each line connecting a polygon vertex is displayed.

Render... (to a window)

Rendering to a window is a good way to "preview" an image before final rendering to a file. Previewing allows you to check the colors, the shadows, the *details* of the image before investing time rendering the final image to a file.

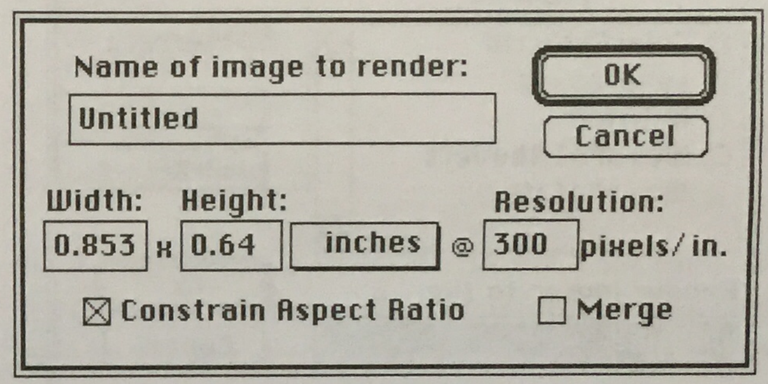
The Render... command spools a file to be rendered to a RenderApp window. While the file gets read into the renderer, the system is "busy." This means that during this time your cursor may seem temporarily stuck, or may move in jerks.

You are first presented with an Open box from which you select a RIB file to be rendered. When you select Open, a second dialog box appears allowing you to set the resolution of the image and the name of the window:



Width and **Height.** These two values determine the resolution or the size of the image. Naturally, larger images take longer to render than smaller ones. But remember that it's really the amount of screen space (the number of pixels) covered by the objects in the image that makes the biggest difference.

Pixels/Inches. When Pixels is selected, this sets pixels as the unit of measurement for the image. When Inches is selected, this sets inches as the unit of measurement for the image. When this is selected, you can also set the resolution, or number of pixels per inch (or DPI):



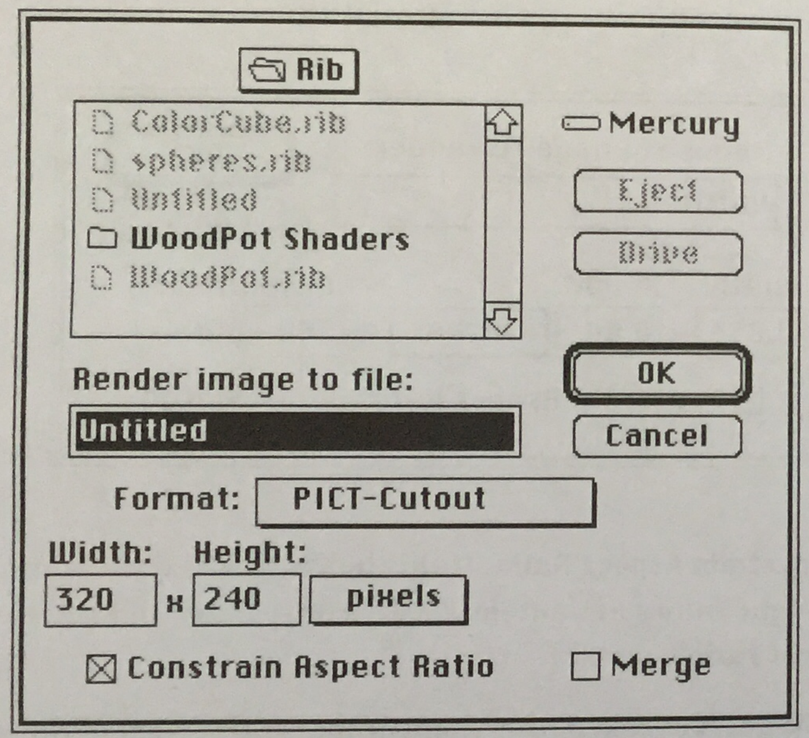
Constrain Aspect Ratio. If this box is checked the width and height values are automatically constrained to keep the aspect ratio constant.

Merge. If checked, this simply renders that part of the image defined by the Crop Window settings into the current RenderApp window, merging it on top of any image already there. Any black background in the new image is considered transparent, allowing the pixels from the original image to show through.

Render To File...

This is the same as the Render entry except that it allows you to specify the location, name and format of a file that is to contain the rendered image.

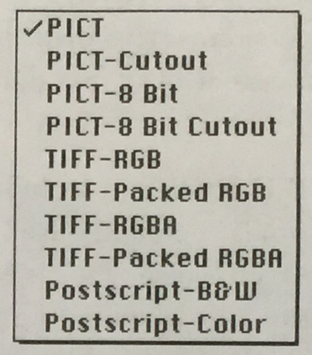
Again, you are first presented with an Open box from which you select a RIB file to be rendered. When you select Open, a second dialog box (see next page) appears allowing you to describe the image file.



The top part of the dialog box allows you to specify the name and folder of the image file. It is a good idea (but not required) to choose file names that give some indication of the file format used (e.g., *img.pict*, or *img.tiff*). If you are rendering several subsections of a larger image (using Crop Windows), give the image files names with numerically sequential suffixes, for example *img.pict 000*, *img.pict 001*, *img.pict 002*, etc. They can then be re-assembled using Open Set (see page 12).

Render image to file. Typing a name here sets the name of the image file.

Format. This pop-up menu specifies the file format of the saved image.



PICT. This creates a 32-bit RGBA PICT file using the name in the name box above. The renderer puts alpha information (commonly used to control transparency or masking in an image) in 8 of the 32 bits.

PICT–Cutout. This creates a 32-bit RGBA PICT file that contains a subset of the whole rectangular image using the name in the name box above. This is usually the smallest rectangle that encloses all the shapes in the image, not the whole image as seen in the display window. Some applications may not use this format as it is intended to be used — the whole rectangular image may appear. The renderer puts alpha information (commonly used to control transparency or masking in an image) in 8 of the 32 bits.

PICT-8 Bit. This is an 8-bit version of the regular PICT format above.

PICT-8 Bit Cutout. This is an 8-bit version of the regular PICT–Cutout format above.

TIFF–RGB. This is probably the most generally useful form of TIFF. No compression is used, and no alpha information is present (see TIFF–RGBA). See the note about TIFF on page 14.

TIFF–Packed RGB. This is the same as TIFF–RGB, but is compressed using PackBits compression, saving disk space. Some software cannot handle compressed TIFF. See the note about TIFF on page 14.

TIFF–RGBA. This is the same as TIFF–RGB, but includes alpha information. Alpha information is commonly used to control transparency or masking in an image. This is especially useful in maintaining smooth edges when saving just the shapes in an image, rather than the whole rectangular image area. See the note about TIFF on page 14.

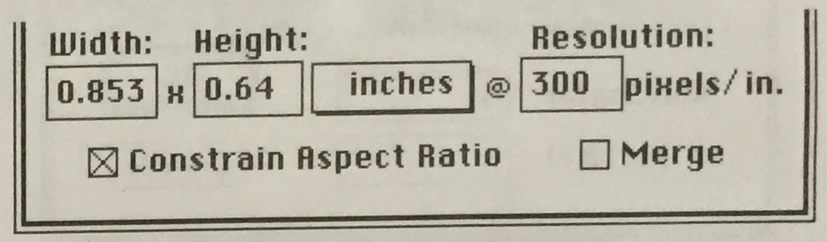
TIFF–Packed RGBA. This is the same as TIFF–RGBA above, but packing is used to save disk space. See the note about TIFF on page 14.

PostScript–B&W. This creates a gray-scale Encapsulated PostScript (EPS) file using the name in the name box. This format should print on both color and non-color printers.

PostScript–Color. This creates a color Encapsulated PostScript (EPS) file using the name in the name box. This format may not be readable by some non-color printers. If it is, it will simply appear in gray-scale.

Width and **Height.** These two values determine the resolution or the size of the image. Naturally, larger images take longer to render than smaller ones. But remember that it's really the amount of screen space (the number of pixels) covered by the objects in the image that makes the biggest difference.

Pixels/Inches. When Pixels is selected, this sets pixels as the unit of measurement for the image. When Inches is selected, this sets inches as the unit of measurement for the image. When this is selected, you can also set the resolution, or number of pixels per inch (or DPI):



Constrain Aspect Ratio. Checking this forces the ratio of the width and height to remain the same. You can set either one and the other will be computed automatically to maintain the image's aspect ratio. When this is not checked, you can create an image of any rectangular shape.

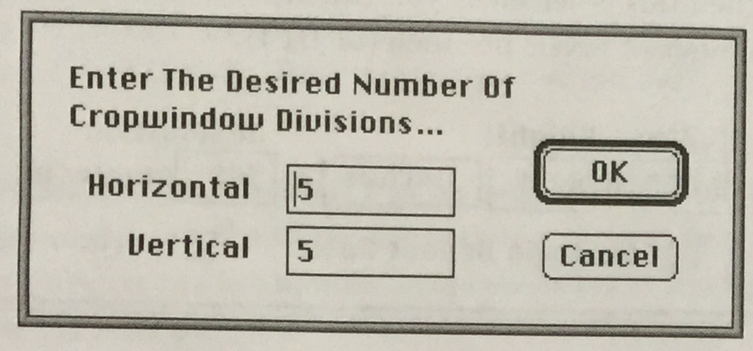
Merge. When rendering to a black and white PostScript file, Merge renders onto a white background instead of a black one. Otherwise, this has no effect; you can't merge part of one file into another.

Rendering large images

You can use crop windows to render images that would ordinarily take more memory than you have available. The idea is to render small areas of a large image, each of which only requires some fraction of the memory required to render the whole image.

To accomplish this you can have RenderApp automatically divide up an image into a set of crop windows. However, it is meant to be used for fairly simple images: there should be no shadows, and the RIB file must contain only one frame. To use this feature:

1. Hold down the Command, Shift, and Control keys while selecting either Render or Render To File. This brings up the following dialog:



2. Set the number of crop windows in each direction and click on OK. A useful rule of thumb for these numbers is to divide the dimension by 400 and round off to the nearest integer (2048 / 400 = 5.12 ≈ 5).

If you're rendering to a window you'll see all the crop windows appear in the same window. If you're rendering to a file, a series of sequentially-numbered image files will get created. You can open the composite image (with all the crop windows in one window) using Open Set from the File menu.

Note: *Be sure you have enough disk space before you start rendering large images!* To calculate this, multiply the image's width times its height (in pixels) by 4. This is the number of bytes you'll need. So, for example, an image that's 2048 by 1536 would require 2048x1536x4, or 12,582,912 bytes (almost 12.6 MB) of disk space. This number will be the same for rendering to screen or to file, but when rendering to screen, you'll need the space to be on the drive specified in the *Pixar Preferences* file. You can see what the default is, or change it by editing this file. It is a plain text file in the System folder's *Preferences* folder. Just look for the line that begins with "/Display/TempFilePath."

3 Rendering RIB Files

This chapter concerns *rendering*, the process of turning the wireframe shapes in a modeling appliction window into a picture by rendering to the screen (previewing) or to a file. The two processes are completely independent. You can preview a scene without rendering it to a file first, and you can render it to a file without seeing what it's like on the screen. In both cases you have control over the trade-off between speed and quality, and you can choose to render part or all of the scene.

Managing Rendering Jobs with RenderMonitor

When you render a preview or render to a file, the modeler itself doesn't actually do the required calculations. It sends this job to a separate application: the renderer. The renderer relates to the modeler in a way that is similar to the way a laser printer relates to a text processor: the renderer is an external resource that processes a job sent to it by an application through an intermediary that monitors jobs. For text processors, this intermediary is the PrintMonitor; for a modeler it is the RenderMonitor.

Like Apple's PrintMonitor, the RenderMonitor is an application with a few well-defined functions. It maintains a list of jobs to be sent on to another application for processing and always sends the one that's at the top of the list to that application. Additionally, it allows you to cancel the current job, remove a job from the list, or delay processing it. The difference is that RenderMonitor sends jobs to a renderer instead of to a printer.

Background Rendering

By default, RenderMonitor runs entirely in the background, silently. When your modeling application sends a file to be rendered, RenderMonitor automatically puts the job at the end of the queue as it's received, processes the queue by sending the files to the renderer specified by the Chooser, and removes the requests from the queue when they are successfully rendered.

This leaves you free to go about using the Macintosh for other work (including, for example, generating more rendering requests).

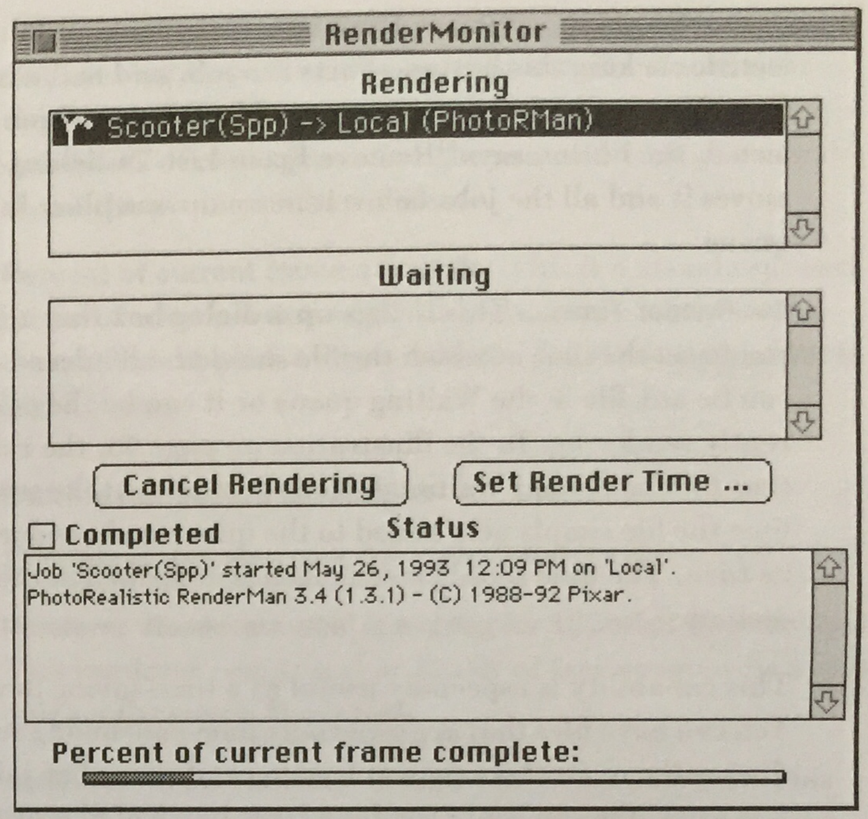
Bringing up the RenderMonitor Window

You can bring up the RenderMonitor window in two ways:

1. Double-click the RenderMonitor icon in the System Folder if there are no jobs in the queue.
2. Pull down the Apple menu and select the RenderMonitor entry while a job is being processed.

You'll see a window something like the one on the next page.

Rendering. The status of a running job appears when you select the job in the "Rendering" list. Status information includes the date and time that rendering was started, on what



machine the rendering is taking place, what type and version of renderer is being used, and any informational or error messages associated with the rendering job.

Waiting. This shows all the jobs waiting in the queue. The status of a waiting job appears when you select it. The first 64 jobs appear in this queue. Later jobs are still queued up; they just don't appear in the scrollable list until there's room to include them. Status information for waiting jobs consists of a list of the renderers that could do that job, along with the status of each renderer (a description of why the renderer is not available). See the "Waiting job status messages" section below for descriptions of status messages for a renderer.

Cancel Rendering/Remove From List. If the current job is selected, clicking this button aborts the job, and makes the first "Waiting" job the current one. If a "Waiting" job is selected, the button says "Remove From List." Clicking it removes it and all the jobs below it move up one place in the queue.

Set Render Time.... This brings up a dialog box that allows you to set the time at which the file should be rendered. This can be any file in the Waiting queue or it can be the one currently rendering. In the illustration on page 36, the render time for the second Waiting job has been set. At the selected time the file simply gets added to the queue and rendered in its turn. The time setting mechanism is described in the next section.

This capability is especially useful as a time-saving device. You can have files that are especially time-consuming render during the night, leaving you free to render smaller jobs as you work. Or you could render a large batch of files over the weekend.

Completed. Checking this box displays the log of the most recently completed jobs in the Status window, along with a message showing the amount of time the job took to render. Otherwise, it displays the status of whatever job is highlighted. You can specify the number of jobs kept in the log in the Preferences dialog, available under the RenderMonitor File menu.

This area may also display the status of jobs which were removed from the Waiting queue but never rendered. See the "Messages displayed when a job wouldn't render" section below for descriptions of messages related to such jobs.

Status. This displays any messages received from the renderer. These may be purely informational (start/end times, CPU time used, etc.), or they may indicate problems in the file, missing elements referenced by the file, insufficient memory, etc. See [err msg file?] for a description of the various kinds of messages you might see, and what you can do about the problems they indicate.

Percent of current frame complete. This is a visual representation of what portion of the current image has been rendered. This applies to the job in the Rendering area that is highlighted.

Waiting job status messages

Possible status messages for a renderer are described below.

Renderer 'Renderername' is waiting for 1234K of free memory. This renderer requires that 1234K of free memory be available on the user's Macintosh.

Renderer 'Renderername' is in use on this Macintosh. This renderer is busy rendering some job in the Rendering list.

Renderer 'Renderername' is available. RenderMonitor is in the process of trying each renderer assigned to the job and has not yet tried this one.

Renderer 'Renderername' is unable to launch. This renderer isn't properly installed or some other condition exists which is preventing the executable file for this renderer from being launched.

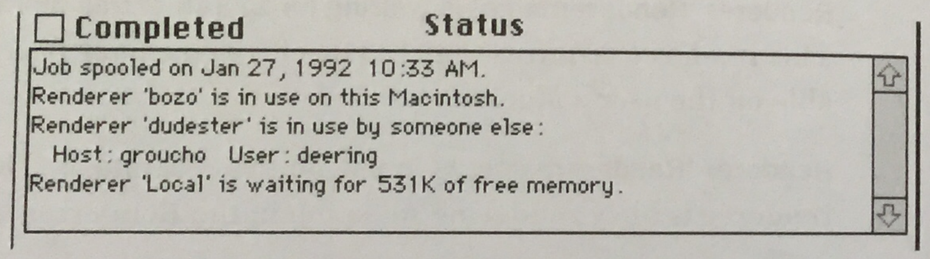
Renderer 'Renderername' is unable to launch due to an internal error. This renderer isn't usable due to some problem with the rendering mechanism.

Renderer 'Renderername' is not responding. This renderer (machine) is not available on the network, the rendering server is not running on the machine, the machine has not

been properly set up in the Chooser, or some other condition exists which prevents the renderer on this machine from being used (NetRenderMan only).

Renderer 'Renderername' is in use by someone else:. This renderer is in use by someone else on the network. When this message is displayed, it is followed by a list of the users which are currently using this renderer (NetRenderMan only).

In the Status area below, the machine "bozo" is already rendering a job from this Mac; the user deering is rendering from his Mac ("groucho") to the machine "dudester;" and the Mac itself needs 531K more memory before it can begin to render any jobs.


Completed
Status
Job spooled on Jan 27, 1992 10:33 AM.
Renderer 'bozo' is in use on this Macintosh.
Renderer 'dudester' is in use by someone else:
Host: groucho User: deering
Renderer 'Local' is waiting for 531K of free memory.

Messages displayed when a job wouldn't render

In addition to displaying the status of recently rendered jobs, the "Completed" status area may also display the status of jobs which were removed from the Waiting queue but never rendered. There are three possible messages to describe why a job couldn't be renderered.

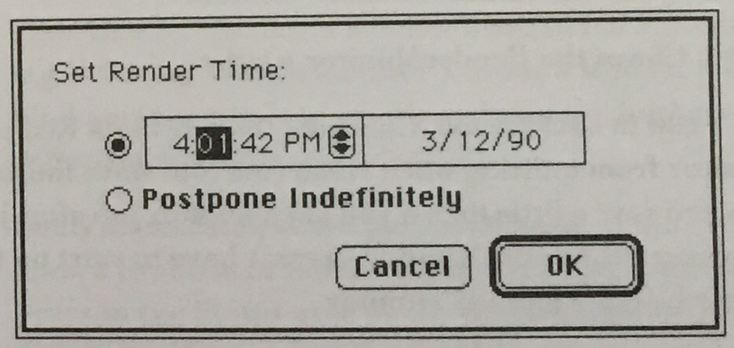
Job {Jobname} couldn't run because no renderers were selected when it was spooled. The current list of renderers in the Chooser does not contain any selected renderers. (For information on how to select a renderer, see the section on "Using the Chooser" on page 47.)

Job {Jobname} couldn't run because no renderers were available. An application spooled a job and did not wish to wait for renderers to become available. When this message is displayed, it is followed by a list of the renderers that were assigned to this job along with the status of each renderer (see the section above describing the possible status messages for renderers in the "Waiting" list.)

Job {Jobname} couldn't run because the spooling application didn't respond. This condition will occur when a job is quickly canceled in an application, before RenderMonitor has had a chance to queue up the job.

Setting the rendering time

To set the rendering time, select a file from those listed in the Waiting area of the RenderMonitor window, or the one currently rendering, and click on Set Render Time....



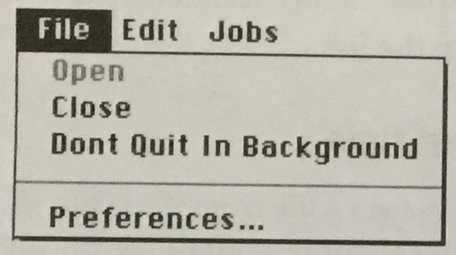
Time/Date button. This is always set to the current time and date unless you change it. Clicking on any of the number fields brings up a pair of arrows to the right. Click on the up arrow to increment the number, and on the down arrow to decrement the number.

Postpone Indefinitely. Setting this option keeps the job in the queue, and waits to send it to the renderer until you click on the Time/Date button.

To confirm your setting, click on OK.

The RenderMonitor File Menu

RenderMonitor's File menu allows you to control the way the RenderMonitor window works.



Open. Opens the RenderMonitor window.

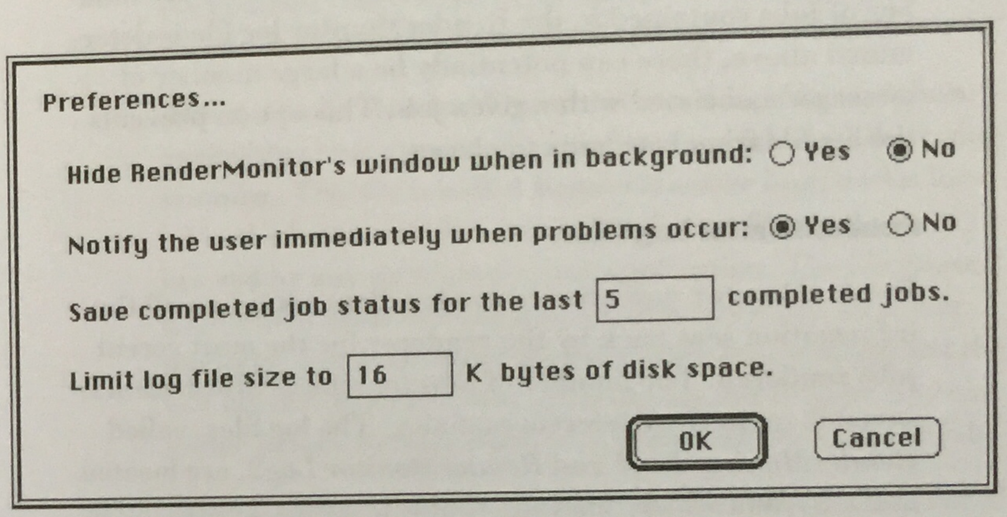
Close. Closes the RenderMonitor window.

Don't quit in background. Checking this prevents RenderMonitor from quitting when rendering jobs have finished. This can save a little time if you keep sending job after job to be rendered; RenderMonitor doesn't have to start up for each job — it's already running.

Preferences.... Opens the Preferences dialog. See the section below for an explanation of this dialog.

The Preferences dialog

This window allows you to control when the window appears, and to set the place for the renderer to look for a special folder.


Preferences...
Hide RenderMonitor's window when in background: Yes No
Notify the user immediately when problems occur: Yes No
Save completed job status for the last 5 completed jobs.
Limit log file size to 16 K bytes of disk space.
OK
Cancel

Hide RenderMonitor's window when in background. Selecting Yes closes RenderMonitor's window when it would be behind another window, or otherwise in the background. It will remain open if you select No.

Notify immediately when problem occurs. If the renderer finds a problem in the file being rendered, a message appears in the Status area of the RenderMonitor window. Additionally, if you select Yes for this option, a dialog box appears with a message that a problem has occurred. If No is selected, or if Yes is selected but the error is relatively minor, your machine beeps once, and the RenderMan icon flashes over the Apple in the menu bar until you bring up the RenderMonitor window.

Save completed job status for the last ... completed jobs. When the "Completed" box is checked in the RenderMoni-

tor window, the status area will contain messages associated with the number of jobs in this box. This text is saved in a RenderMonitor log file.

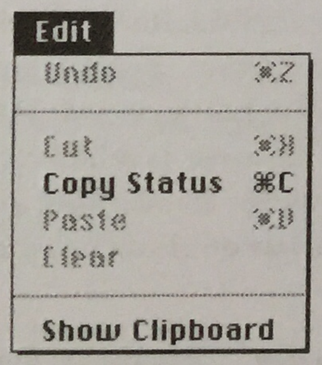
Limit log file size to ... K bytes of disk space. While the number of jobs contained in the RenderMonitor log file is determined above, there can potentially be a large number of messages associated with a given job. This option prevents the log file from becoming too large.

RenderMonitor Log Files

RenderMonitor now creates two log files containing all the information sent back by the renderer for the most recent jobs rendered. The number of jobs for which information is saved is set in the Preferences dialog. The log files, called *RenderMonitor Log1* and *RenderMonitor Log2*, are located in the System folder. They are simple text files, and so can be opened by any word processing application.

The RenderMonitor Edit Menu

Since you can't really edit anything in RenderMonitor, most of its commands are grayed out. However, this menu does provide one important function.



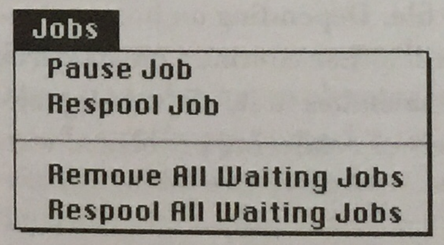
Copy Status. This copies the entire contents of the Status area of the RenderMonitor window and places it on the Clipboard. You can then open a text editing application and paste in the contents of the Clipboard. In this way you can permanently save a record of any error messages you've encountered, or a list of all the images you've rendered.

Alternatively, RenderMonitor creates a log file containing everything that appeared in its Status window while it was running. This file is called *RenderMonitor Log*, and is located in the System folder. It is a simple text file, and so can be opened by any word processing application. The file doesn't get created until RenderMonitor actually exits, so if you queue up a new job and then remember that you wanted the last log file, you need only grab the file out of your System folder before RenderMonitor is done with the current job.

Show Clipboard. This simply displays the current contents of the Clipboard.

The RenderMonitor Jobs Menu

The Jobs menu provides control over all the jobs in the RenderMonitor window.



Respool Job. This makes it convenient to assign a new renderer to a job after it has already been spooled. Just select a new renderer in the Chooser, then select Respool Job. The

job will be sent to the newly selected renderer (when it becomes available).

Pause/Resume Job. Stops the rendering process immediately. MacRenderMan keeps track of where it left off rendering and starts exactly at that place when you resume (unless you reboot the machine in the meantime). Once you've started rendering, the memory used by the renderer is unavailable to other applications, even while you pause. So if you try to launch another application and get a "Not enough memory..." error, pausing the renderer won't free up any memory.

Respool All Waiting Jobs. This simply respools every job that's not yet rendering. Use this to respool a large number of frames you've sent to be rendered, for example.

Remove All Waiting Jobs. This simply cancels every job that's not yet rendering. Use this to cancel a large number of frames you've sent to be rendered, for example.

Problems reported by RenderMonitor

When a message appears in the Status area of the RenderMonitor window it usually means the renderer has found something wrong in the file. Depending on how serious the problem is, rendering will either continue on and produce an image that may have mistakes in it, or just stop altogether. So there are two levels of rendering problems: warnings and errors.

Warnings

A warning means that there's something wrong with the RIB being rendered, but rendering can continue. The image may or may not look wrong; in some cases no image will appear at all. Usually, there is an easy fix for this sort of problem.

If you have the RenderMonitor window preferences set to "Notify immediately when problem occurs," when the renderer encounters a minor problem a warning box appears saying "An error has occurred:" followed by a description of the error. You must acknowledge the warning by clicking on the OK button. In addition, warnings appear in the status area of the RenderMonitor window.

Errors

Errors cause the renderer to send a message to the RenderMonitor window with information about the problem, and then to give up completely on the file. However, if there are more jobs in the queue they will be processed normally.

A description of all the errors you're likely to encounter and what to do about them is in two plain text files in the release: *MRM Errors Part 1*, and *MRM Errors Part 2*, in the *Release Notes* folder.

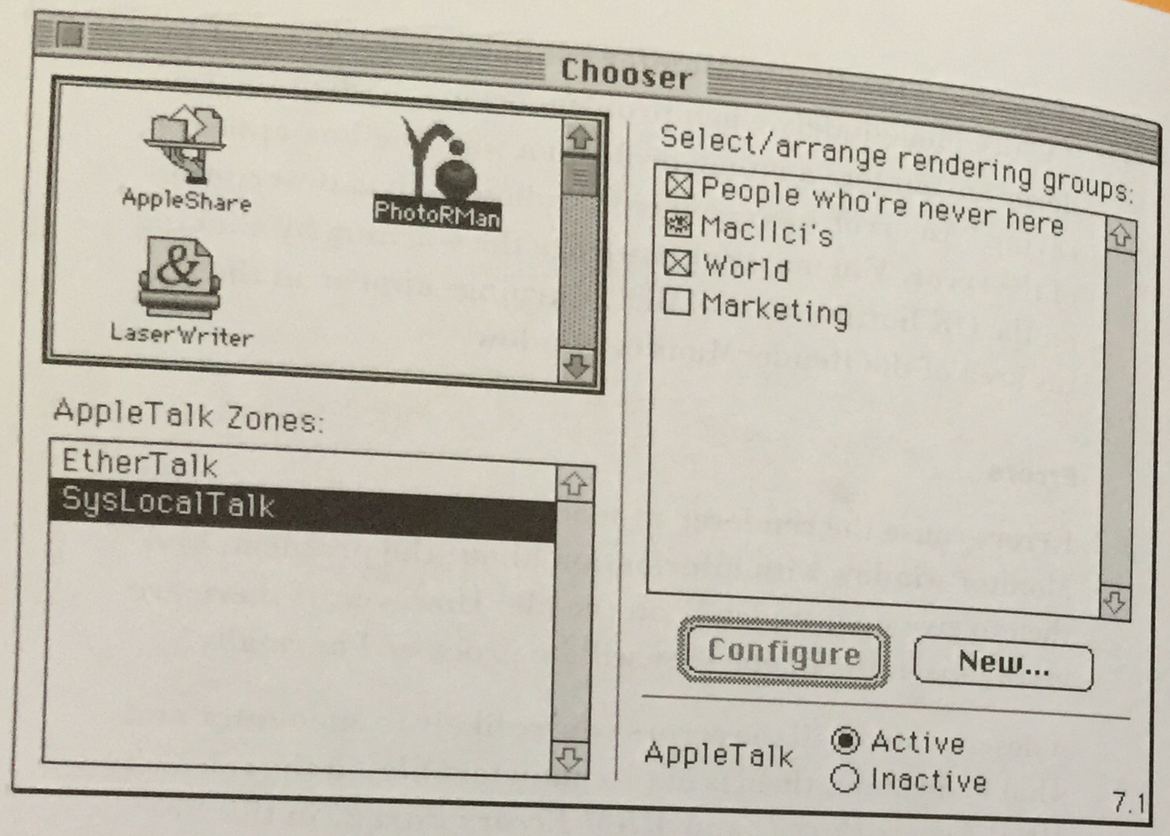
Using the Chooser

The Chooser has some added capabilities when MacRenderMan is installed. These are detailed in this section.

If you have access to more than one renderer (you might have a rendering accelerator board, for example) you'll need to pick one to render on. You do this in a way that's similar to the way you choose a Printer: by using the Chooser.

To select a renderer:

1. Pull down the Apple menu and select "Chooser."
2. Click on the PhotoRMan icon. The Chooser will display any available renderers.



3. Click on the name of the renderer you want to use (usually you'll have only your "Local" renderer).
4. Click on the checkbox next to the name to make the renderer available for rendering.
5. Click and drag the renderer to be tried first up to the top of the list.

Renderer names in the Chooser have a checkbox to the left of them. This box can have one of three things in it:

- Nothing (☐). The group is not selected for rendering at all.
- Regular check (☒). The renderer is selected for rendering. If the renderer is not busy, the next job in the RenderMonitor queue will go to it.

- Eyeball (▩). The renderer is selected for preview rendering only. Any jobs queued in RenderMonitor will not be sent to the renderer, making it always immediately available for previewing. To make this selection, hold down the Option key while clicking in the box.

Having a renderer only render previews is especially useful. This allows you to render to a file in the background while you use a different renderer for preview rendering.

In the Chooser setup above, "Local" is selected for normal rendering; "Spanky" is selected for preview rendering only.

For normal rendering, regular checked renderers are tried in order. Renderers reserved for previewing are ignored. If all regular checked renderers are busy, the job waits in the queue for the first *available* checked renderer, regardless of its place in the list.

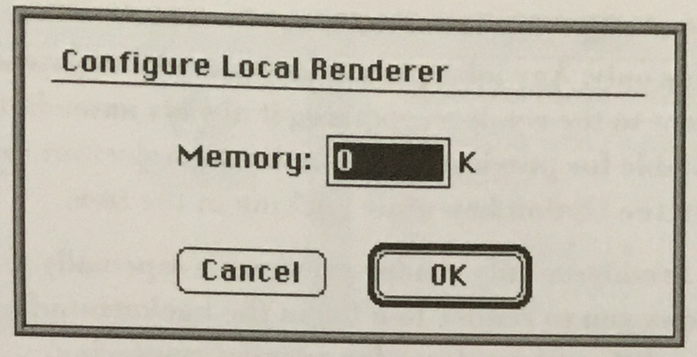
For preview rendering, renderers marked with an eyeball are tried in order. If they are all busy, renderers marked with a regular check are tried in order. If all renderers are busy, the job waits in the queue for the first *available* renderer, regardless of its place in the list or reserved status.

Arranging the order of renderers

Renderers are used in the order in which they appear in the Chooser. To change a renderer's position, simply click and drag the name up or down the list.

Renderer configuration options

Clicking the Configure button brings up a dialog box that allows you to set the amount of memory allocated to the selected renderer. Allocating more memory will allow you to render larger or more complex images. You should have at least



4 or 5 megabytes of memory just for the renderer. Fairly complex scenes will require more like 8 MB. It would be a rare image that required as much as 14 MB. To get a good estimate of the largest value you should use for the system you have, follow these steps:

1. While in the Finder, choose About the Finder... from the Apple menu and make a note of the number in "Largest Unused Block."
2. Use Get Info from the File menu to look at the "Application Memory Size" for RenderMonitor and any other application (e.g., RenderApp) that isn't currently running, but will be while you're rendering.
3. Add up all the numbers you got from the Get Info windows above.
4. Subtract the total from the "Largest Unused Block" number from step 1 above. This is the largest number you should use in the Renderer Memory box.

If you're using virtual memory, the procedure is the same. You'll just see more memory in the About the Finder window. If you're using the Vector RenderMan renderer, you may be able to reduce its memory allocation and still operate efficiently.

You should note that the value you get in step 1 may change from time to time, as it can be affected by memory fragmentation if you've started and closed many programs since booting your Macintosh. Nevertheless, this should provide an idea of the largest value you can allocate for rendering.

If you'd rather not go through this calculation and want to configure the renderer so that it always takes the largest amount of memory available, enter 0 for the Renderer Memory. The catch here is that you will be giving the renderer permission to take up all available memory, which will probably prevent you from starting any new applications after a rendering has started. You will either have to be sure to start all the applications you need before rendering starts, or calculate a more reasonable value to type in the Configure box.

APPENDIX:

Editing RIB Files

We hope you never need to edit a RIB file. However, you may find that sometimes you need to make some small changes to one. There are often only a few things you might want to change or add in a RIB file:

Header information. This contains the renderer settings: quality, size, output file type, etc.

Light information. This defines the light sources: intensity, color, shadow-casting capability, location, etc.

Surface information. This is a shader call, and sets whatever parameters are appropriate for the shader.

Texture creation. This is usually a one-line RIB file to convert a TIFF or BMP file to a texture map suitable for use by a shader.

The sections below contain a little information on these topics to get you started, but you can get the complete scoop from two sources (both available from Pixar):

- *The RenderMan Interface, Version 3.1.* This specification completely defines the RenderMan Interface at the lowest level. It includes definitions of the standard RenderMan shaders, a summary of RenderMan Shading Language syntax, RIB file conventions, and more.
- *The RenderMan Companion.* This 475-page book is the only book available on RenderMan itself. It provides a conceptual background, an interface description, programming examples, sample images, and an introduction to advanced rendering concepts.

RIB Syntax

For our purposes, RIB commands consist of a one-word command, which may be followed by one or more arguments. Each argument is followed by a value enclosed in brackets. Arguments can be a string in double quotes, a value, or a set of values. Here are a few examples:

```
AttributeBegin
Surface "wood" "grain" [5] "texturename" ["bali.tex"]
Sphere 1 -1 1 360
```

In the course of editing RIB, you do some typing. If the rendered image doesn't come out the way you expected, go back and very carefully check what you typed. All quote marks and parentheses must be matched, and since RIB is case-sensitive, be careful of capitalization changes. Also beware of text editors that replace straight quote marks ("rgba") with curved ones (“rgba”).

Typing mistakes can cause error messages that may not lead you to suspect typos as the cause, though the image may render more or less as it should. Or the renderer may just quit after sending error messages. One strategy for pinpointing the cause is to find the line in the RIB file referenced by the very first error message and look on the line before it for typos or syntax errors.

Header Information

Header information, which sets the renderer's controls, is usually found at the very beginning of a RIB file. However, this can also be a separate file that gets sent to the renderer just before a file with only scene information. The most common parameters define the image size in pixels, amount of detail, and sharpness:

```
Format 256 192 1
ShadingRate 1
PixelSamples 2 2
```

Light Information

This is often near the beginning of a RIB file. Be sure that each LightSource call has a unique light number (the first number in the line). For a spotlight, the from and to parameters do what you'd expect. But for a distant light they merely define a direction, *not a place*. If there is a falloff parameter, be sure it's turned up high enough for the light to reach the surface you need to illuminate.

```
LightSource "ambientlight" 1 "intensity" [.1]

LightSource "distantlight" 2 "intensity" [0.8]
            "from" [4 4 -3] "to" [0 0 0]
            "shadowname"["distlight.shd"]
```

Surface Information

This call can have any number and combination of parameters. The first parameter after the Surface call is the name of a shader (minus its .slo extension if it has one).

```
Surface "carpet" "Ka" [1] "texturename" ["borobudur.tex"]
```

Texture Creation

You can convert an image file to a texture with a simple 1-line RIB file. The structure is:

MakeTexture call
input image name
output texture name
x-direction abyss behavior
y-direction abyss behavior
filter type
x-direction filter size
y-direction filter size

Abyss behavior defines what happens on the surface outside the area covered by the image. Periodic simply repeats the image over and over; clamp smears the last line of pixels across the surface; black fills the rest of the surface with pixels with a value of 0.

The filter types are the same as those in the Custom Quality dialog, as are the filter size controls. See the descriptions of these in the section on "Render Setup... using PhotoRealistic RenderMan" on page 23.

For example:

```
MakeTexture "ballpic.tiff" "ballpic.tx" "periodic" "clamp"
            "box" 1 1
```


Be sure to use the texture map's exact file name in the Surface call that uses it (and don't forget the quotes):

```
Surface "paintedplastic" "texturename" ["ballpic.tx"]
```

A

abyss behavior, 56
aliasing, 26
allocating memory for renderers, 49–51
Alpha channel, 16–17, 18
    See also background
"An error has occurred" messages, 47
antialiasing, 26
appearance of surfaces, 3
Apply Gamma Correction checkbox, 16
arguments in RIB commands, 54
aspect ratio
    constraining, 29, 33
    of pixels, 26

B

background of images
    changing color, 18
    using same background for several images, 14
    viewing Alpha channel, 17, 18
    See also Alpha channel
background rendering, 6, 36, 42
    See also RenderMonitor
black
    abyss behavior, 56
    filling area with, 18
Box pixel filter, 25
brightness (gamma correction), 16, 19
bucket size for rendering, 25

C

calculating image size, 34
camera. *See* viewer
canceling
    pausing rendering, 46
    postponing render time indefinitely, 42
    removing waiting jobs, 46
    rendering, 38
Cancel Rendering button, 38
Channels command, 17, 20
Chooser, 47–51
    arranging order of renderers, 49
    default renderer, 7
    overview, 6
    renderer configuration options, 49–51
    selecting a renderer, 47–49
clamp abyss behavior, 56
Clear command, 18
Clipboard
    copying RenderMonitor Status area to, 45
    viewing, 18, 45
Clip To Alpha Channel checkbox, 16
Close command
    RenderApp, 13
    RenderMonitor, 42
closing
    RenderApp windows, 13
    RenderMonitor window, 42

closing *(continued)*
See also quitting
color
changing for background, 18
dithering, 16, 19
EPS file format, 32
exposure setting, 26
matching screen colors to image file, 20
for wireframe images, 27
color display, 7
Completed checkbox, 38, 43–44
compressing image files, 15
Configure button, 49
Constant shading interpolation, 25
constraining aspect ratio, 29, 33
Copy Status command, 45
Crop Window
clearing to black, 18
dividing large images, 33–34
merging into RenderApp window, 27, 29
resetting to defaults, 23
setting for rendering, 22–23, 26
customizing MacRenderMan, 9

D

darkness (gamma correction), 16, 19
date, setting for rendering, 41
defaults
drive for saving rendered images, 34
renderer, 7
render settings, 22
resetting for Crop Window, 23
resetting for Render Setup dialog, 23, 26
shading rate for rendering, 24
depth setting for image files, 15
detail, amount in images, 24, 55
disk space for images, 17, 34
displaying. *See* viewing
distant lights, 55
Dither checkbox, 16
Dither command, 16, 18, 19
dithering images, 16, 18, 19
dividing large images, 28, 33–34
Don't Quit In Background command, 42

E

editing RIB files, 53–57
header information, 53, 55
light information, 53, 55
reference books, 54
RIB command syntax, 54–55
surface information, 53, 56
texture creation, 53, 56–57
typing mistakes and, 54–55
Edit menu (RenderApp), 17–18
Clear command, 18
illustrated, 17
Merge Over Color command, 18
Show Clipboard command, 18
Edit menu (RenderMonitor), 44–45
Edit menu (RenderMonitor) *(continued)*
Copy Status command, 45
illustrated, 44
Show Clipboard command, 45
Encapsulated PostScript (EPS) formats, 32, 33
"An error has occurred" messages, 47
Esoterica button, 24
Esoterica dialog box, 24–26
exiting. *See* quitting
exposure for rendering, 26

F

falloff, 55
file formats, 14–15, 31–32
File menu (RenderApp), 12–17
Close command, 13
illustrated, 12
Merge File command, 14
Open command, 12
Open Set command, 12–13
Save As command, 13
Save command, 13
Save dialog box, 14–17
Save Selection As command, 13
File menu (RenderMonitor), 42–44
Close command, 42
Don't Quit In Background command, 42
illustrated, 42
Open command, 42
Preferences command, 42–44
files. *See* image files
filters for pixels
setting type, 25
size of, 25
filter types for textures, 56
formats
for rendering images, 31–32
for saving images, 14–15

G

Gamma command, 16, 18, 19
gamma correction, 16, 19
Gaussian pixel filter, 25
geometry of objects, 3
grid size for rendering, 25

H

header information in RIB files, 53, 55
height of rendered image, 29, 32
hidden-surface elimination, 26
hiding RenderMonitor window when in background, 43

I

image files
compressing, 15
converting to textures, 56–57
formats, 14–15, 31–32
naming, 13, 30–31
opening, 12
opening sets, 12–13
rendering to, 17, 30–33
saving, 13, 14–17
See also RIB files
Image menu, 18–21
Channels command, 17, 20
Dither command, 16, 18, 19
Gamma command, 16, 18, 19

Image menu *(continued)*
illustrated, 18
Optimize Colors command, 18, 20
Show at Normal Size command, 21
Show at Smallest Size command, 21
Zoom In command, 21
Zoom Out command, 21
image sets. *See* sets of images
image size
calculating, 34
display, 11
image subsections
dividing large images, 28, 33–34
rerendering, 22–23
saving, 13
See also Crop Window; sets of images
immediate notification when problems occur, 43
inches as unit of measurement, 29, 33

J, K

jaggies, 26
Jitter checkbox, 26
"Job {Jobname} couldn't run . . ." messages, 40–41
Jobs menu, 45–46
illustrated, 45
Pause/Resume Job command, 46
Remove All Waiting Jobs command, 46
Respool All Waiting Jobs command, 46
Respool Job command, 45–46

L

large images, rendering, 33–34
light information in RIB files, 53, 55
lighting, modeler description of, 3
LightSource calls, 55
limiting RenderMonitor log file size, 44
log files (RenderMonitor), 44, 45
LZW compression, 15

M

MacRenderMan
customizing, 9
diagram of rendering process, 8
overview of applications, 6–7
reference books, 54
See also specific applications
MakeTexture calls, 56–57
mask, Alpha channel as, 16–17
memory, renderer requirements, 49–51
Merge File command, 14
Merge Over Color command, 18
merging
Crop Window into RenderApp window, 27, 29
PICT or TIFF images with current image, 14
white background for PostScript image, 33
messages
immediate notification when problems occur, 43
RenderMonitor warnings and errors, 46–47
waiting job status, 39–40
when job wouldn't render, 40–41
See also troubleshooting
modelers
basic concepts, 3–4, 5
defined, 2
modeling, 2
monitors, color, 7

N

names/naming
image sets, 13
saving images using different name or location, 13
when rendering to file, 30–31
notification when problems occur, 43

O

objects, modeler description of, 3
Open command
RenderApp, 12
RenderMonitor, 42
opening
image files, 12
image file sets, 12–13
RenderMonitor window, 36
Open Set command, 12–13
Optimize Colors command, 18, 20

P

PackBits compression, 15
Pause Job command, 46
Percent of current frame complete, 39
periodic abyss behavior, 56
PhotoRealistic RenderMan renderer
Alpha channel, 16–17
overview, 7
Render Setup using, 23–26
PhotoRMan. *See* PhotoRealistic RenderMan renderer
PICT-8 Bit Cutout format, 31
PICT-8 Bit format, 31
PICT-Cutout format, 31
PICT files
compressing, 15
depth setting for, 15
displaying and saving, 7
formats, 31–32
merging on top of current image, 14
saving images as, 14–17
saving without dithering, 16
Pixar Preferences file, 9
pixels
aspect ratio, 26
bucket size for rendering, 25
filters for rendering, 25
grid size for rendering, 25
number of bits for (depth), 15
resolution, 29
sampling rate for rendering, 24
as unit of measurement, 29, 33

Postpone Indefinitely button, 42
PostScript-B&W format, 32, 33
PostScript-Color format, 32
preferences
dialog box, 43–44
Pixar Preferences file, 9
Preferences command, 42–44
Preferences dialog box, 43–44
preview for rendering, 28, 49
printing, rendering compared to, 8–9
problems. *See* troubleshooting

Q

queue for rendering
applications managing, 6, 7
canceling/removing jobs, 38
displaying completed jobs, 38, 43–44
overview, 8–9
postponing render time indefinitely, 42
removing waiting jobs, 46
RenderMonitor Waiting list, 37
respooling jobs, 45–46
setting render time, 38, 41–42
waiting job status messages, 39–40
See also RenderMonitor
Quit command, 14
quitting
RenderApp, 14
RenderMonitor, 42
See also closing

R

RAM, renderer requirements, 49–51
refresh number for rendering, 27
Remove All Waiting Jobs command, 46
Remove From List button, 38
RenderApp, 11–34
Edit menu, 17–18
File menu, 12–17
Image menu, 18–21
overview, 7, 11
quitting, 14
RenderApp window, 11–12
Render menu, 21–34
RenderApp window
closing, 13
illustrated, 12
merging Crop Window into, 27, 29
overview, 11–12
rendering to, 28–29
"Renderer 'Renderername' is . . ." messages, 39–40
renderers
arranging order of, 49
basic concepts, 4
configuration options, 49–51
default renderer, 7
defined, 2
memory requirements, 49–51
preview renderer, 49
respooling jobs to, 45–46
selecting, 6, 47–49
rendering
in background, 6, 36
canceling, 38
rendering *(continued)*
Crop Window for, 22–23, 26
default drive for saving images, 34
defined, 2
diagram of process, 8
dividing large images, 28, 33–34
to file, 17, 30–33
formats for files, 31–32
large images, 33–34
modeling vs., 2
pausing, 46
postponing render time indefinitely, 42
previewing, 28, 49
printing compared to, 8–9
queue, 6, 7, 8–9
rerendering image subsections, 22–23
RIB files, 35–51
setting time for, 38, 41–42
setup for PhotoRMan, 23–26
setup for VectRMan, 26–28
to window, 28–29
wireframe images, 7
The RenderMan Companion, 54
The RenderMan Interface, 54
RenderMan Interface, defined, 2
RenderMan Interface Bytestream protocol files. *See* image files; RIB files
RenderMan Shading Language, 4
Render menu, 21–34
default settings, 22
illustrated, 21
Render command, 28–29
rendering large images, 33–34
Render Setup command, 21–22
using PhotoRMan, 23–26
using VectRMan, 26–28
Render To File command, 30–33
Reset Crop Window command, 23
Set Crop Window command, 22–23
RenderMonitor
background rendering, 36
overview, 6, 35–36
RenderMonitor window, 36–42
warnings and errors, 46–47
RenderMonitor log files, 44, 45
RenderMonitor window, 36–42
Cancel Rendering button, 38
Completed checkbox, 38, 43–44
illustrated, 37
messages
waiting job status, 39–40
when job wouldn't render, 40–41
opening, 36

RenderMonitor window *(continued)*
Percent of current frame complete, 39
Rendering list, 36–37
Set Render Time button, 38, 41–42
Status list, 38–39
Waiting list, 37
See also queue for rendering
Render Setup command, 21–22
for PhotoRMan, 23–26
for VectRMan, 26–28
Render Setup dialog box, 21–22
resetting to defaults, 23, 26
using PhotoRMan, 23–24
using VectRMan, 26–28
Reset Crop Window command, 23
resolution
when rendering to file, 32
when rendering to window, 29
Respool All Waiting Jobs command, 46
Respool Job command, 45–46
Resume Job command, 46
RIB files
command syntax, 54–55
defined, 2
editing, 53–57
header information, 53, 55
light information, 53, 55
reference books, 54
rendering, 35–51
surface information, 53, 56
texture creation, 53, 56–57
See also image files

S

sampling rate for rendering, 24, 55
Save Alpha Channel checkbox, 16
Save As command, 13
Save command, 13
Save dialog box, 14–17
Save Selection As command, 13
saving
Alpha channel, 16
completed job status, 43–44
current image, 13
disk space required, 17
dithered images, 19
gamma-corrected pixels, 16
image subsections, 13
images using different name or location, 13
Save dialog box, 14–17
tips, 17
scenes, modeler description of, 3–4, 5
selecting
renderers, 6, 47–49
See also opening
Set Render Time button, 38, 41–42
sets of images
dividing large images, 33–34
naming, 13, 30–31
opening, 12–13
saving, 13
shaders, basic concepts, 4, 5
shading
default rate, 24
defined, 4
interpolation, 25
Show at Normal Size command, 21
Show at Smallest Size command, 21
Show Clipboard command
RenderApp, 18
RenderMonitor, 45
Smooth shading interpolation, 25
speed of rendering
bucket size and, 25
Crop Window and, 22–23
divisions and, 28
refresh setting and, 27
resolution and, 29, 32
VectRMan wireframe images and, 7, 27, 28
spotlights, 55
Standard Hider setting, 26
Status list, 38–39, 45
subsections of images. *See* image subsections
surface information in RIB files, 53, 56
syntax of RIB commands, 54–55

T

Tagged-Image File Format. *See* TIFF files
texture creation using RIB files, 53, 56–57
texture maps, 7, 56–57
TIFF files
compressing, 15
depth setting for, 15
formats, 32
format variations, 14–15
MacRenderMan unable to read, 14–15
merging on top of current image, 14
saving images as, 14–17
TIFF-Packed RGBA format, 32
TIFF-Packed RGB format, 32
TIFF-RGBA format, 32
TIFF-RGB format, 32
time, setting for rendering, 38, 41–42
Time/Date button, 41
transparency information. *See* Alpha channel
troubleshooting
aliasing or jaggies, 26
editing RIB files, 54–55
immediate notification when problems occur, 43
messages
RenderMonitor warnings and errors, 46–47
waiting job status, 39–40
when job wouldn't render, 40–41
not enough disk space to save image, 17
RenderMonitor problems, 46–47
TIFF file unreadable by MacRenderMan, 14–15

U

unit of measurement, 29, 33

V

Vector RenderMan renderer
  overview, 7
  Render Setup using, 26–28
VectRMan. *See* Vector RenderMan renderer
viewer, modeler description, 3
viewing
  Alpha channel, 17, 18
  channels, 20
  Clipboard, 18
  completed rendering jobs, 38
  gamma correction, 16, 19
  image as small as possible, 21
  matching screen color with image file, 20
  normal size image, 21
  RenderMonitor Waiting queue, 36–37
  zooming, 11, 21
virtual memory, 50–51

W, X, Y

Waiting list, 37
  See also queue for rendering
white background for PostScript images, 33
width of rendered image, 29, 32
windows
  closing, 13
  dividing large images, 33–34
  Optimize Colors and, 20
  RenderApp window, 11–12
  rendering to, 28–29
  See also Crop Window; RenderApp window; RenderMonitor window
wireframe images
  color for, 27
  rendering, 7, 26–28
  See also Vector RenderMan renderer

Z

Z Buffer Hider setting, 26
zoom box (RenderApp window), 11
zoom factor display, 11
Zoom In command, 21
Zoom Out command, 21

UNITED STATES POSTAL SERVICE®

a preferred shipping service on ebay

08/19/2019
From 55810
0 lbs 5 ozs
Zone 6

US POSTAGE PAID
Pitney Bowes
ComBasPrice
NO SURCHARGE

024P0007776135

USPS FIRST-CLASS™ PKG

Wayne Cook
222 1st Avenue S
Proctor MN 55810-2323

0000

9327230063394

P · I · X · A · R

1001 West Cutting Boulevard
Richmond, CA 94804

510 236 4000

Installing MacRenderMan

Hold on a sec!

Before you charge off and do the installation we'd like to draw your attention to one little detail. This applies only if you want to put the application files on a disk other than your boot drive. If this is the case, be sure that in Step 3 below you select your boot drive. That's all.

Default installation

If you're installing all the software on your boot drive you'll need 3.4MB of disk space. Once the installation is complete you'll get about 0.9MB back.

If you'd like to install the software on a remote drive (hard disk, Syquest, optical, etc.), you'll still need almost 3MB on your boot drive for installation purposes. There are some files that get put in the boot drive's System Folder. These take up about 2.2MB. So after the installation is complete, you'll get 0.8MB back on your boot drive. The files on your remote drive will take up about 0.5MB.

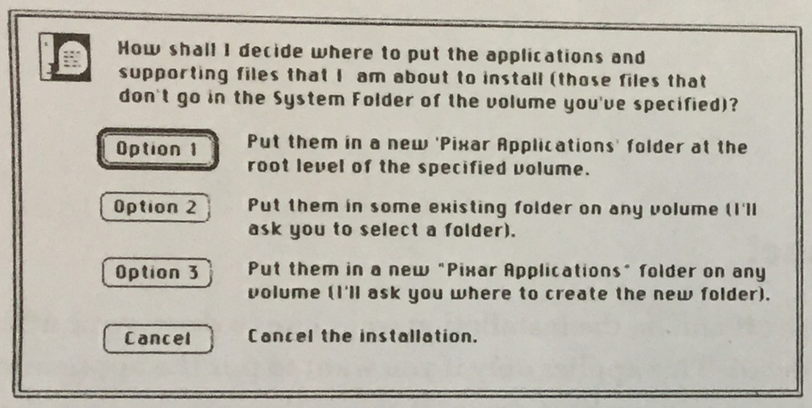
To do the default installation:

1. Insert Disk 1; double-click on the Installer. This brings up a splash screen.

2. When the splash screen comes up, click on OK. This brings up the installer screen.

3. In this screen, the boot drive is usually selected by default. The only reason you would use a different setting is if you plan to boot off another drive. In this case, use the Switch Disk button to select a different boot drive.

4. Click on the Install button. This brings up a dialog asking where to put the rest of the files (the ones that don't go in the *System Folder*).


How shall I decide where to put the applications and supporting files that I am about to install (those files that don't go in the System Folder of the volume you've specified)?
Option 1
Put them in a new 'Pixar Applications' folder at the root level of the specified volume.
Option 2
Put them in some existing folder on any volume (I'll ask you to select a folder).
Option 3
Put them in a new "Pixar Applications" folder on any volume (I'll ask you where to create the new folder).
Cancel
Cancel the installation.

5. Click on one of the options. This will begin the installation. As the installation proceeds, compressed files will get copied to your system, and then get decompressed.

6. When the installation is done a dialog will appear. Click on Quit to quit the Installer.

7. After the installation is done, be sure to restart your system.

Custom installation (installing options)

There are a number of data files, system files, application files, Read Me files, and so on in this package. All the components that you can select in the installer are listed on the next page.

When installing an option, allow an extra 1.4MB on your boot drive for space for decompression.

Option	Installed size	Description
MacRenderMan	2400K	RenderMan renderer, render spooling software, shaders
NetRenderMan (Unix)	600K	Software for rendering on a Unix machine across a network
NetRenderMan (Mac)	500K	Software for rendering on 3 Macintosh machines across a network

To install one of the custom options, first do steps 1–3 of the default installation. Then:

3a. Click on Customize. This brings up the custom installation dialog.

3b. Choose one or more of the Options. Continue on to step 4 of the default installation.

If you already have MacRenderMan...

If you already have a version of MacRenderMan, it will not be overwritten by this installation process. This means that you will have both the old and the new version of some software.

Some parts of the old MacRenderMan package are absent from the new version. These are ShaderApp, the Pencil Demo, source code for the shaders, and the contents of the *Samples* folder, all located in the *MacRenderMan* folder. If you've been using these you might want to keep them around. Other parts get updated automatically. These are all in the *System Folder*. The main new thing that gets updated is RenderApp, now located in your new *Pixar Applications* folder.

UNITED STATES POSTAL SERVICE®

a preferred shipping service on ebay

F

08/19/2019
From 55810
0 lbs 5 ozs
Zone 6

US POSTAGE PAID
Pitney Bowes
ComBasPrice
NO SURCHARGE

024P0007776135

USPS FIRST-CLASS™ PKG

Wayne Cook
222 1st Avenue S
Proctor MN 55810-2323

0000

9361 2300 6394

P • I • X • A • R

1001 WEST CUTTING BLVD.
RICHMOND, CA 94804
TEL: 510/236-4000 ◆ FAX: 510/236-0388